Lahden ammattikorkeakoulu
Lahti University of Applied Sciences

# GANTRY ROBOT

## With linear motor technology

LAHTI UNIVERSITY OF APPLIED SCIENCES
Mechatronics Engineering
Gantry Robot Thesis
25.05.2017
Autor: Doral Vall, Judith
Tutor: Lahtinen, Teijo

Lahti University of Applied Sciences
Degree Programme in Mechatronics Engineering

DORAL VALL, JUDITH: Gantry Robot
With linear motor technology

Bachelor's Thesis in mechatronics

Spring 2017

# Abstract

L’objectiu d’aquest projecte és la programació d’un robot capaç de dibuixar amb el suport de SEW_EURODRIVE. Aquest projecte forma part d’un altre més extens que consisteix en el disseny, la selecció de components i la construcció. Aquest projecte principal esta dividit en tres part principals: la mecànica, elèctrica i electrònica. En aquest treball s’inclou la part electrònica on es tracten temes com la configuració, el control, la automatització i la programació.

**Paraules clau: robot cartesià; SEW_Eurodrive; configuració.**

El objetivo de este proyecto es la programación de un robot apto para dibujar con el soporte de SEW_EURODRIVE. Este proyecto forma parte de otro más extenso que consiste en el diseño, la selección de componentes y la construcción del robot. Este proyecto principal está dividido en tres partes: mecánica, eléctrica y electrónica. En este trabajo se incluye la parte electrónica donde se tratan temas como la configuración, el control, la automatización y la programación.

**Palabras clave: robot cartesiano; SEW_Eurodrive; configuración.**

The aim of this project is to program a drawing robot with the support of SEW_EURODRIVE. This project is part of a more extensive one that consists of the design, the selection of components and the construction of the robot. This main project is divided into three parts: mechanical, electrical and electronic. This work includes the electronic part where topics such as configuration, control, automation and programming are covered.

**Key Words: gantry robot; SEW_Eurodrive; configuration.**

# Contents

## Acknowledgments

First, I would like to begin by thanking my family the opportunity to be studying here, at LAHTI University of Applied Sciences, Finland.

I would like to thank Mr. Teijo Lahtinen for giving me the opportunity to participate in this work and for his support throughout their development and I would also like to thank Mr. Jaako Lajunen of SEW Eurodrive who has supported me in the configuration and programming section of the robot.

I am very grateful for this opportunity because this way I learned new programming languages, I met the company SEW Eurodrive, which I loved, and I have been able to work on something that really fascinates me: the automation and robotics.

# 1. Introduction

## 1.1. Project Aims

The first thing anyone can think about when taking a look at this project is… what role does it have? What is it for? At first, the work consists of a Cartesian robot, also called Gantry robot, which are usually used for pick and place tasks but they can also be used in welding and other applications.

The main objective of this project is the realization of a program that allows the robot to draw eights with a linear motor technology and move their axis.

Apart from the main objective, one of the aims of this project is to be able to deal with possible technical problems that may occur when continuing with a project of another team of people. Therefore, it will require a good understanding of the machinery and a good configuration of this.

## 1.2. Motivation

Due to the constant advancement of new technologies, especially in electronics, robotics and automation, I would like to be able to learn how to analyse, manage and optimize an industrial production line. Moreover, I am very interested in the automation of industrial processes, as well as production management and programming of industrial robots, so when I saw this project I didn't hesitate in wanting to participate.

Furthermore, this project gives me the opportunity to work with material of an international organization, SEW Eurodrive, which I had never worked with. It also gives me the opportunity to put into practice all my project and work with a team of local people, which could learn new working methods and new ways of doing things and practicing and improving English.

## 1.3. Scope of the project

This is a big project, which includes a high range of different types of works. As it is aforementioned, another group of students have been responsible for the design of the robot, the selection of components, the construction of this and its electrical installation.

The electronical part is the one of this project where the configuration of the components have to be properly done to be able to run a program. The programming and configuration of the machine have been done with the software MoviTools MotionStudio from SEW Eurodrive.

## 1.4. Theoretical framework

### 1.4.1. Definitions

To begin, here we have the definition of robotics according to different sources of information:

- According to Spanish Real Academy (RAE), robotics is a technique that applies computing to the design and use of devices that, in substitution of people, perform operations or jobs, usually in industrial facilities.

- According to Webster, robotics is a technology dealing with the design, construction, and operation of robots in automation.

As it is seen above, robotics can be defined as a discipline dedicated to the study, design, realization and use of robots.

In the same way, the word “robot” is defined below:

- According to Spanish Real Academy (RAE), robot is a machine or electronic programmable element, capable of manipulating objects and performing operations previously reserved for people only.

- According to Webster, a robot is a real or imaginary machine that is controlled by a computer and is often made to look like a human or animal.

As it is seen, robot could be defined as an automatic device designed to perform some of the functions traditionally performed by humans.

### 1.4.2. Types of robots and robotics

The robots can be classified in many ways, here is presented a type of classification in which the following types are disaggregated:

- Linear/Gantry Robot: are Cartesian coordinate robots whose three principal axes are linear. They are usually used in industrial environments. The robot of these project is of this type.

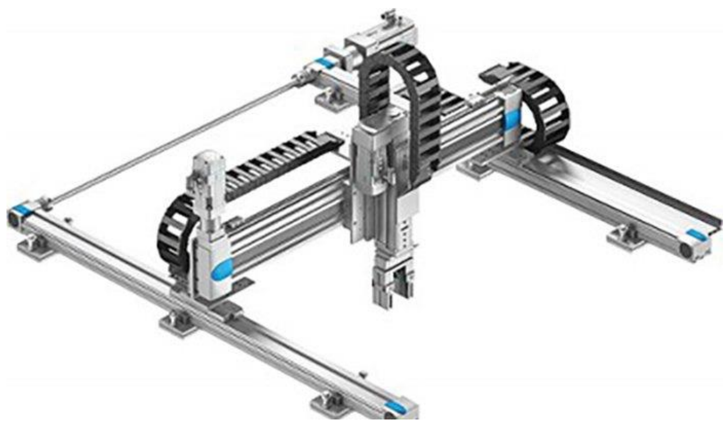


***Image 1.-*** *Gantry Industrial Robot.*

- Manipulator Robots: are mechanical articulated structures with manipulation capacity which are normally used in automated production environments, although there are applications outside the industrial environment.


***Image 2.-*** *Example of a manipulator robot application: KUKA KR-1000.*

- Mobile Robots: These are robots with mobility that are usually based on wheels. This type of robot is used in very different environments.


***Image 3**.- Example of a mobile robot application: AGV.*

- Humanoid robots: robots that present a certain human appearance. It has multiple possible uses as an aid to humans.


***Image 4.-** Example of a humanoid robot: HONDA Asimo.*

- Submarines: is a watercraft capable of independent operation underwater.


***Image 5.-** Example of submarine: MIT Submarine.*

- Drones: An unmanned aerial vehicle (UAV), is an aircraft without a human pilot aboard. It have a lot of applications nowadays.

***Image 6.-** Example of drone: PNG Drone.*

There are a lot of other types of robots with different applications.

Nowadays, robotics could be classified in two big application groups:

- Industrial Robotics: Is born of practical demands of industrial production. Is a key element in flexible automation, aimed at reducing costs.

- Services Robotics: encompasses all types of applications in which robots provide services to humans outside the industrial environment: domestic robotics, assistance robotics, entertainment robotics...

### 1.4.3. History of industrial robotics

It can be said that the history of industrial robotics begins in 1948 when R.C. Goertz, of the Argonne National Laboratory, developed a master-slave mechanical system for the telemanipulation of radioactive elements. It wasn’t until 1954 that the original mechanical transmission was replaced by electric transmission with servocontrol.

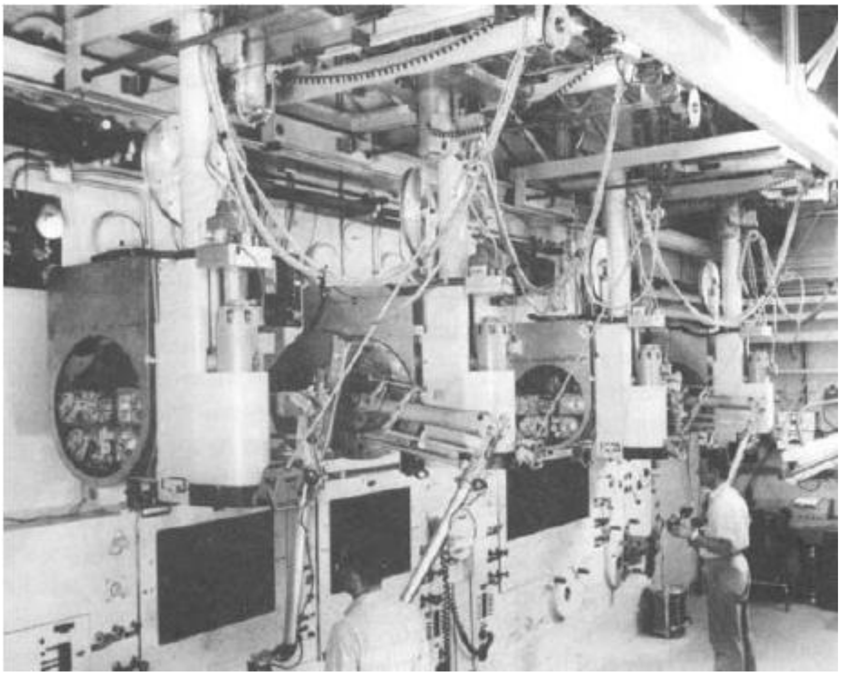


***Image 7.-*** *Goertz telemanipulators.*

In 1954 Georg C. Devol developed the idea of a "programmed article transfer machine" and in 1956 partnered with Joseph F. Engelberger to found Unimation Company with which in 1959 they developed Unimate robot which, with 2 tons of weight and with Hydraulic actuation, reproduces trajectories in joint space. In 1961 they installed their first Unimate at the GM factory in Trenton.

***Image 8.-*** *Unimate, the first industrial robot.*

In 1960 industrial acceptance began when other companies, such as AMF, produce the Versatran. As a commercial hook, this type of machine was called "robot". Versatran was the first cylindrical robot that was installed in a Ford factory. In 1969 Unimation installed the first welding robots at the GM plant.

From that moment on, there is a market expansion of robotics and automation. Events like the following demonstrate this stage. In 1968 J.F. Engelberger visits Japan and signs agreements with Kawasaki for the production of Unimate for the Asian market. In 1970, the German company KUKA installed one of the first lines of welding equipped only with industrial robots.

Since then, robotics spread in universities with the creation of the first artificial intelligence laboratories at MIT, Stanford and Edinburgh in 1964.

Then begins the era of electric robots:

- In 1974 the Swedish company ASEA, nowadays called ABB, markets the IRB6. This was the first commercial robot with control by microprocessor and driven purely electric.

- In 1978 Unimation / Vicarm with support of GM commercialize the robot PUMA, of which acronym means: Programmable Universal Machine for Assembly, based on the designs of Schinman.

- In 1981 at Carnegie Mellon developed the concept of direct action, without reducers, that allows faster and more precise movements.

At this point, it begins to develop and create one of the most important robots in the industry: the robot SCARA. In 1978 Hiroshi Makino, from Yamanashi University, developed the SCARA robot, which stands for Selective Compliance Assembly Robot Arm, low cost and assembly-oriented configuration. In 1979 the firsts SCARA robots were marketed by Sankyo, Pentel and NEC. It wasn't until 1984, when Adept introduced the first SCARA robot with direct action, called AdeptOne.


***Image 9.-*** *SCARA Robot.*

The advance of industrial robotics with SCARA robots was parallel at the advance at the use of artificial vision. In 1981 GM developed CONSIGHT, which was the first artificial vision system capable of locating and identifying pieces on a conveyor belt. In 80s, the professor Raymond Clavel of EFPL, Switzerland, developed the Delta robot, which is a parallel robot designed for the quick manipulation of small and light objects. Based on the design of Clavel, in 1998 ABB marketed the FlexPicker.

At present, new designs and advances in the world of robotics and automation are focusing on dual-arm architecture, in robots with integrated vision system and in easily programmable robots. Great progress is also being made on robots that can collaborate directly with humans.

### 1.4.4. Actual economic and social benefits of robots

Although there are many different futuristic theories about robots and their benefits or harms, there are three clear benefits nowadays:

- Quality: the robots ensure a high and uniform quality of the final product. This is due to the high level of repetitiveness of the tasks performed by the robots.

- Security: The robots minimize the presence of people in dangerous manufacturing processes, reducing the chances of accidents at work and replacing tedious work operators.

- Robots replace humans in dull, dirty and dangerous tasks.

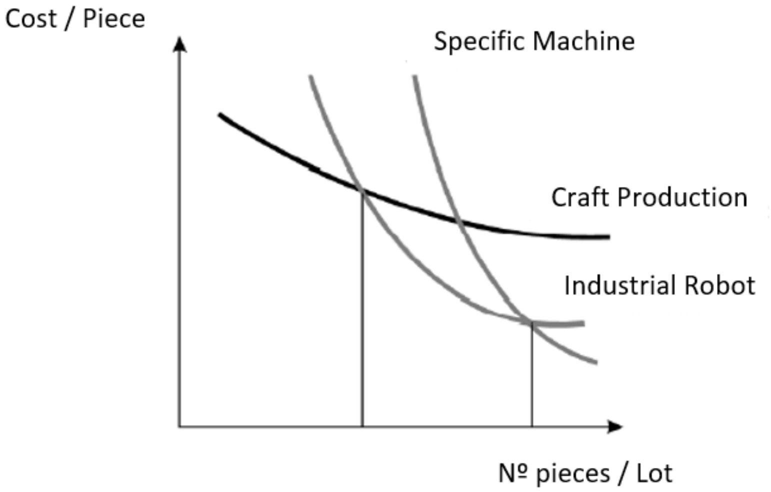



***Image 10.-*** *Economical benefits of industrial robots.*

### 1.4.5. Statistics of the use of robots

To begin with, the topic is introduced with sales data and global operating stock. In 2013, 178.132 robots were sold and the global operating stock stands at around 1.600.000 units (assuming 15 years of useful life time for each robot).

Below is a graph of World Robotics 2014 (IFR) where the estimated worldwide annual shipments of industrial robots is represented:


***Image 11.-*** *Estimated worldwide annual shipments of industrial robots.*

Regarding to the regions with more robots, according to a study of 2013, Asia (including Australia and NZ) is the largest robot owner with 55% of global acquisitions. Next is Europe with 24%, followed by America with 17% of the acquisitions of robots worldwide.

Asia is not only the region with the most automatic activity, it's also the one with the highest growth rate in terms of robotization.

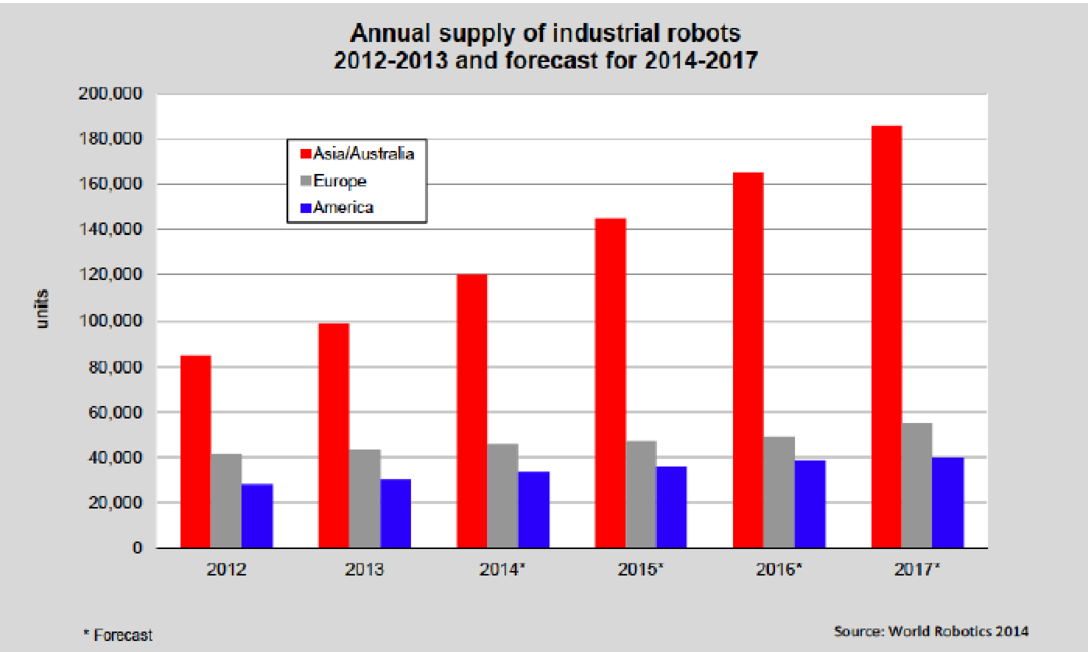



***Image 12.-*** *Annual supply of Industrial robots by World Robotics 2014.*

By sales in 2013, the largest markets are:

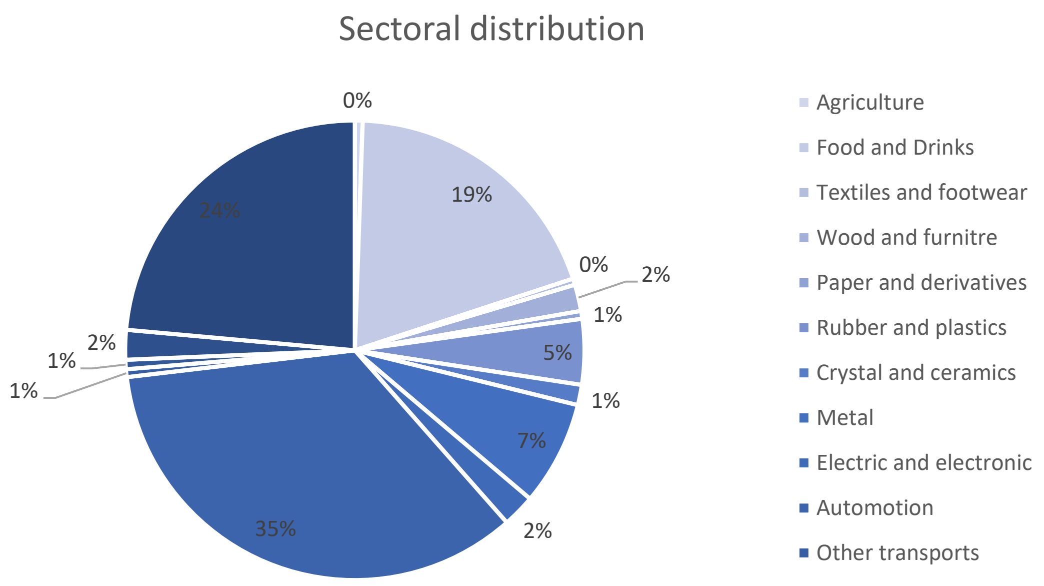
- China with 36.560 units representing 20% of the total.
- Japan with 25.110 units representing 14% of the total.
- United States with 23.700 units representing 13% of the total.
- Korea with 21.300 units representing 12% of the total.
- Germany with 18.300 units representing 10% of the total.

The five aforementioned countries were around 70% of the total sales of that year. China has been the fastest growing market for several years and in 2013 it became number one.

In accumulated operating stock, Japan remains the number one with 304.000 units, followed by the United States with 192.000 units, Germany with 167.000 units, Korea with 156.000 and China with 133.000 units.

As for the use of robots depending on the sectors of use we find that the leading sector is the automotive, with the largest park and the highest number of annual sales.

The food and beverage sector is in second place but it is growing more.



## 2. Project Robot

The robot of this project is a gantry robot with linear motor technology. That means that is a kind of Cartesian robot which have three axes of freedom perpendicularly oriented at each other. As you can understand, this ensures a working envelope in a form of a rectangular box.

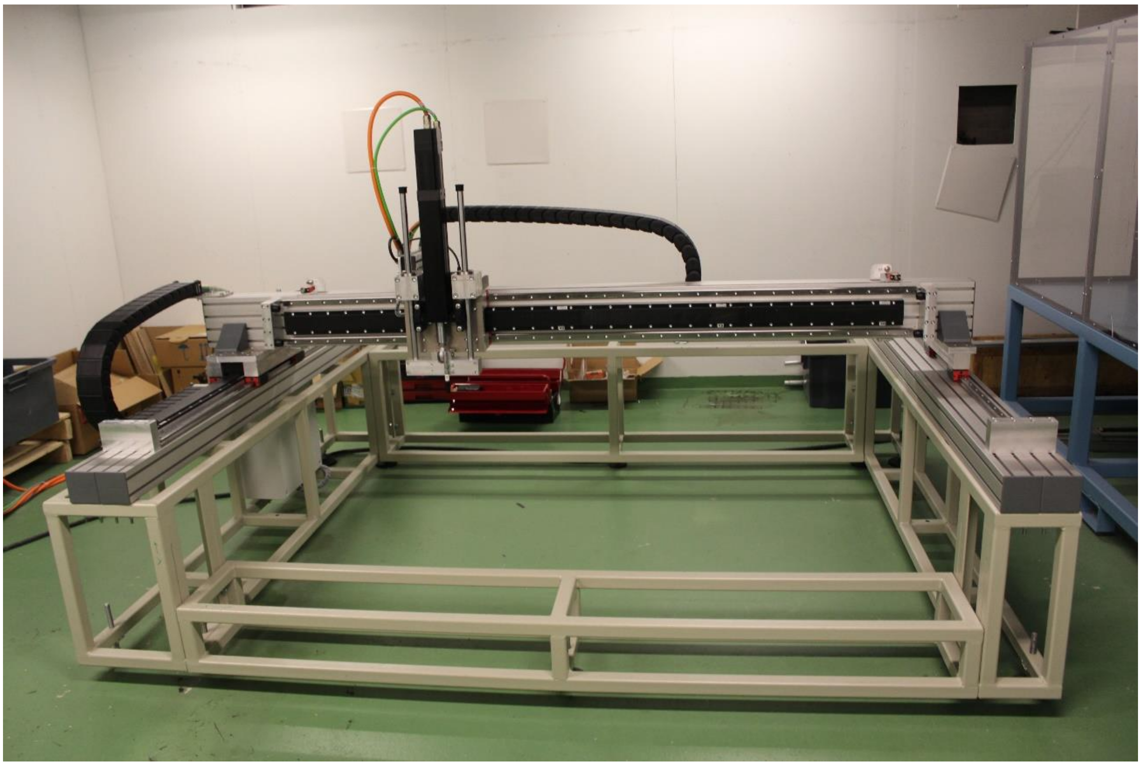


***Image 13.-*** *Gantry robot of the project.*

As you can see in the image above, movement along the x axis takes place between two beams which are directed in the x direction. The carriage on the y axis can move along it between aforementioned x axis beams. The tool can be lowered down from the carriage, thus forming the z axis movement.

The work envelope is similar with other robots of Cartesian type, however, a gantry robot usually encloses its work envelope from the outside. Also, the only part of the robot that interferes with its work space is its z axis and the tool.

As you can see this robot stands firmly on four "legs". If those legs are strong enough, the robot can lift very heavy weights. This is the most prominent use of a gantry robot. So, Gantry robots can be used for pick and place tasks or it can be used as a platform for other robots. For example, a 6-axes robotic arm can be mounted upside-down as a tool on a gantry robot's Z axis.

# 3. Component information

## 3.1. Electronical components

The electronic part of a machine is that one within it can be distinguished the microcontroller, which is the brain of the engine, the sensors, which are the ones that allow to detect the environment, and the power drivers, which are the circuits in charge of supplying the necessary energy to the motors to make them roll.

All the electronic components such as the aforementioned will be explained here.

### 3.1.1. Inductive sensor

#### *3.1.1.1. Some information: inductive sensor*

The inductive sensors incorporate an electromagnetic coil that is used to detect the presence of a conductive metal object. This type of sensor ignores non-metallic objects and are used mainly in the industry, for positioning applications and for detecting the presence of metallic objects in certain contexts.

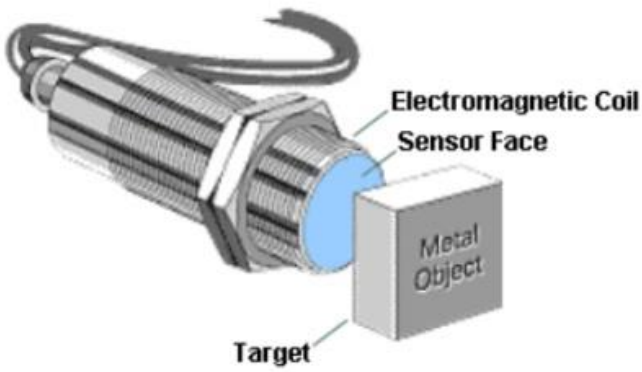



***Image 14.-*** *Inductive Sensor.*

Detecting broken drill bit, detecting set screws on hub for speed or direction control, detecting presence of can and lid, detecting full open or closed valve position, detecting broken drill bit on milling machine are some of the uses of an inductive sensor in an industrial context.

Inductive sensors use the Foucault current operation principle.

The oscillator circuit produces an AC voltage which, when it's applied to the coil, causes an electromagnetic field.

### *3.1.1.2. Operation Principle*

The operation of the Inductive Sensor is mainly based on the oscillator stage:

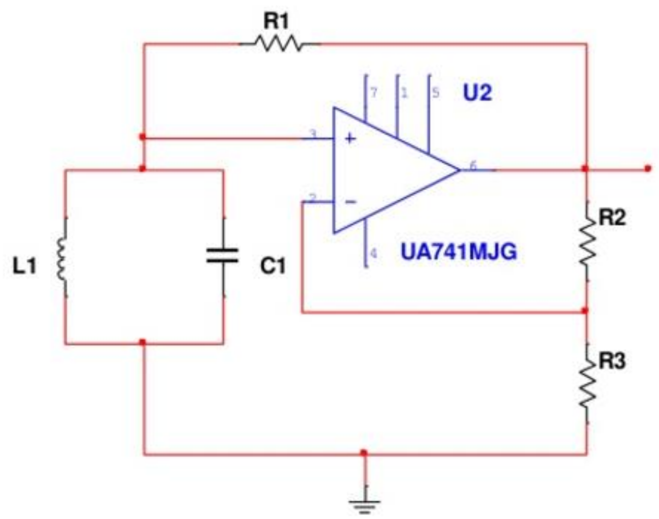



***Image 15.-*** *Oscillator circuit.*

The main function of this circuit is to generate a sinusoidal signal, although it also functions as a high selectivity filter.

The LC circuit generates a sinusoidal frequency signal:

$$f = \frac{1}{2\pi\sqrt{LC}} \tag{1}$$

When a metallic object enters the field, the Foucault currents circulate inside the plate and these generate at the same time a magnetic field opposite to the generated by the oscillator. The reduction of the magnetic field has the effect of reducing the inductance of the coil, changing the frequency of the oscillations.

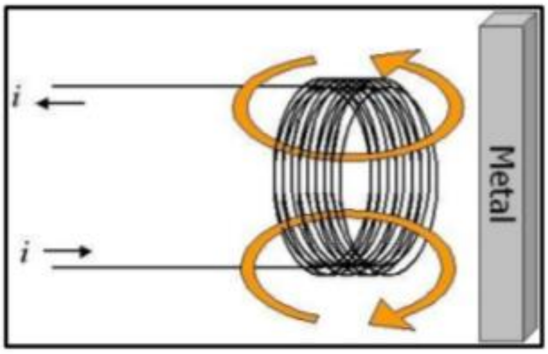



***Image 16.-*** *Foucault corrents.*

As it is aforementioned, the circuit also acts as a high selectivity filter. At small variations in the oscillation frequency the amplitude of the signal decays rapidly.

#### 3.1.1.2.1. Absence of metallic objects

The sinusoidal signal has a frequency that corresponds to the central frequency of the filter. The output signal is equal in amplitude of the input signal, which is the signal generated by the oscillator.

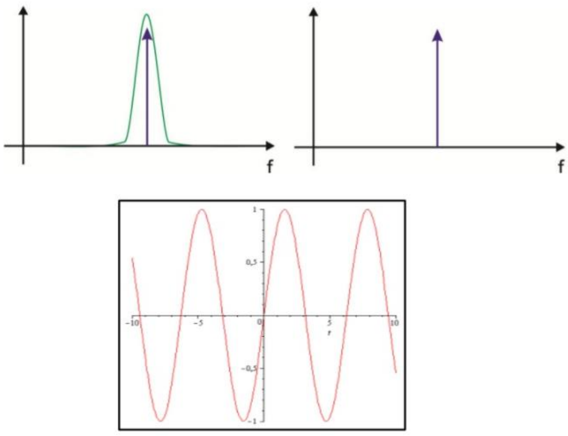



***Image 17.-*** *Sensor signal with absence of metallic objects.*

#### 3.1.1.2.2. Presence of metal objects

The generated sinusoidal signal has a higher frequency than the central frequency of the filter, because when approaching a metallic object to the sensor has the effect of decreasing the inductance of the coil and consequently increases the frequency of oscillations.

The filter greatly attenuates the amplitude of the input signal.

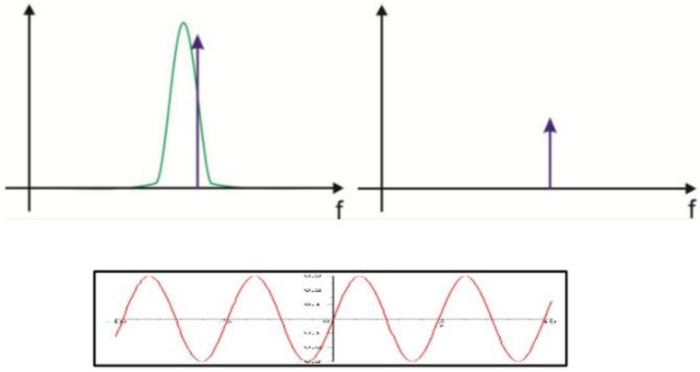



***Image 18.-*** *Sensor signal in presence of metallic objects.*

That way, the amplitude of the oscillator signal is decremented in presence of metallic objects and increased in absence of these.

#### *3.1.1.3. Inductive sensors on the project*

The inductive sensor which is used in the project is the model IF5539 of IFM whose most outstanding characteristics are the following:

- Particularly short housing.
- For use where space is restricted.
- Very high switching frequency.
- Wide operating temperature range.
- Robust metal housing for use in harsh industrial environments.

***Image 19.-*** *Inductive Sensor IF5539.*

On the project there are four inductive sensors: two on the limits of the X axis and two in the Y axis. This sensors are used as a limit switches in order to prevent the shock between the mechanical stops and the motor.

## 3.1.2. Mechanical switches

### *3.1.2.1. Definition*

Limit switches are electromechanical devices consisting of an actuator mechanically linked to a set of contacts. When an object comes into contact with the actuator, the device operates the contacts to close or open an electrical connection.

### *3.1.2.2. Operation*

These sensors have two types of operation: positive mode and negative mode.

In positive mode the sensor is activated when the element to be controlled has a task that causes the axis to rise and connect with the moving object with the NC contact (normal closed). When the spring (pressure spring) breaks, the sensor is disconnected.

The negative mode is the inverse of the previous mode, when the controlled object has a projection that pushes the axis down, forcing the cup spring and causing the circuit to close.

### *3.1.2.3. Final switches models*

Within the final switches sensor devices there are several models:

- Honeywell safety: This limit switch is incorporated within the Honeywell company GLS range and is also manufactured in miniature, in both metal and plastic and wood, with three very compact metal pipes.

- Limit switch for dangerous environments: This is a microswitch with a robust aluminum housing. This cover has been designed to withstand internal explosions and to be able to cool the gases generated by the explosion inside. This switch is actuated by an external roller lever actuator that allows a 360 ° adjustment.

- Set crews: These types of limit switches are used to prevent sensor damage caused by the sensed object. They are composed of a threaded cylinder containing a spring with a metal target which is detected by the inductive sensor.

#### 3.1.2.4. Mechanical switches on the project

On the project there are four mechanical switches: two on the end of the X axis and two in the Y axis. The main function of this switches is to stop the robot before the aluminium plates shock with the aluminium stoppers.

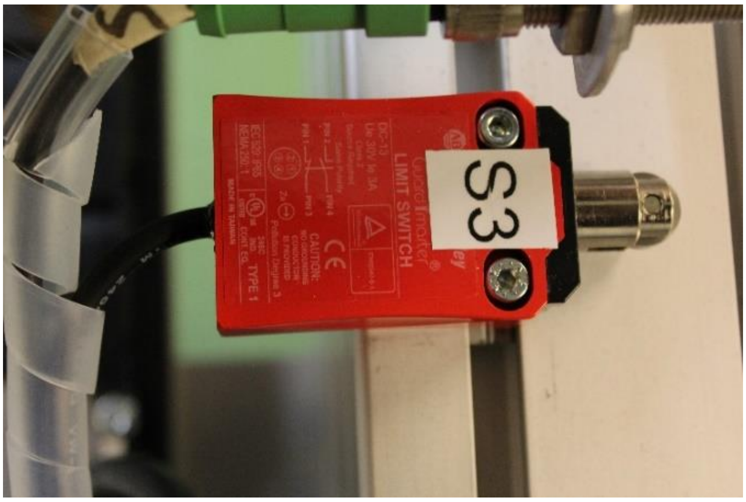



***Image 20.-*** *Mechanical switch of the project.*

They are safety components to control a system. When the robot touch them they break the electrical connection so the machine stops. These are the second security measure that has been taken on the machine. While the inductive sensors stop the axis where it has been activated, the mechanical switches stops the whole machine.

### 3.1.3. PLC

#### *3.1.3.1. Definition*

A programmable logic controller (PLC), or programmable controller is an industrial digital computer which has been ruggedised and adapted for the control of manufacturing processes, such as assembly lines, or robotic devices, or any activity that requires high reliability control and ease of programming and process fault diagnosis.

The objective is to control machines and processes.

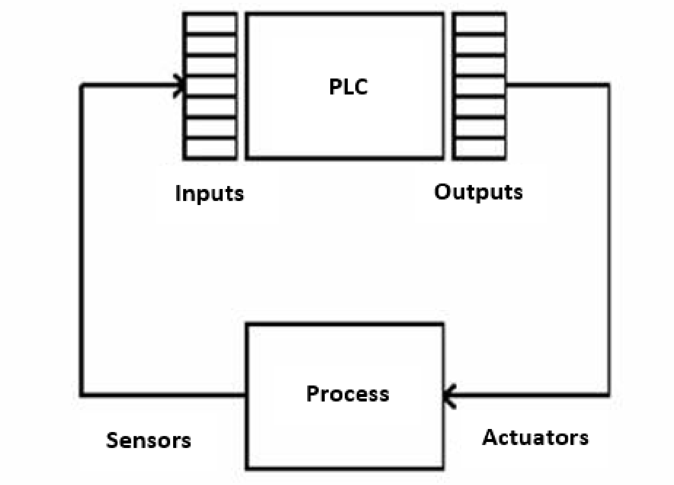



***Image 21**.- PLC basic principles.*

#### *3.1.3.2. Internal Architecture*

A programmable system consists of:

- <u>Central Processing Unit</u> (CPU): is the processor of the system. It executes the instructions. The central processing unit is the one which executes the program performed by the user. It relates the outputs according to the state of the inputs, so that it controls a certain process.

- <u>Memory</u> (MEM): is the part of the system that stores the instruction sequence and data. Is responsible of store the operating system, application programs, intermediate and final results processed by the CPU, as well as the state of the inputs and outputs.

- Input / Output Block (E/S): part of the system that allows communication with the outside data/systems.

    - Inputs: The input module is the link between the PLC and the elements distributed in the field (sensors, pushbuttons...), responsible for collecting information on the current state of the process.
    - Outputs: This module is responsible to establish the link between the PLC and the process actuators. This, takes the results from the CPU and adapts them to appropriate electrical levels that can be used by the actuators.

  The input/output system is connected to the components which will work on the real process; these components can be analogue or discrete. It is usual to isolate the I/O with electromagnetic relays in order to protect the PLC of overloads (usually max voltage in PLC is 24 v).

- Bus: interconnection between the rest of subsystems.

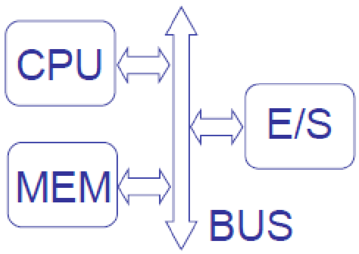



***Image 22.-*** *Internal Arquitecture of a programable System.*

### 3.1.3.3. Motion

The signals arriving from the physical components such as sensors, pushbuttons, limit switches… are weird to the terminals of the input interfaces. Then the CPU with the memory read this data, execute the program and write some other data in the output interface which will control components such as motors, solenoid valves, leds…

### 3.1.3.4. Types of PLCs

PLCs can be classified according to their characteristics in:

- Nano PLCs: Generally it is a compact type PLC, which means that it integrates the power supply, CPU and inputs/outputs. It can handle a reduced set of inputs and outputs, generally less than 100. This PLC allows you to handle inputs and digital outputs and some special modules.

- Compact PLCs: These PLCs have incorporated the power supply, the CPU and the input and output modules in a main module. It can handle from a few inputs and outputs to several hundred. Its size is superior to Nano-type PLCs and they support a great variety of special modules, such as:

    - Analog inputs and outputs
    - Quick counter modules
    - Communication modules
    - Operator interfaces
    - Input and output expansions

- Modular PLCs: These PLCs are composed of a set of elements that make up the final controller. These are:

    - Rack
    - Power supply
    - CPU
    - Input and output modules

  These kind of PLCs exist from the Micro-PLCs, which support a large number of inputs and outputs, to the high performance PLCs that allow to handle thousands of inputs and outputs.

##### *3.1.3.5. Benefits of using PLCs*

The PLC has a lot of advantages over the relays, so they are modular and flexible, fact that allows an expansion of the software and hardware elements when it is required. Furthermore, the units are interchangeable, so it is possible to change one of them for another one bigger in case of lack of memory or input and output terminals. Some other important benefits apart of the ones commented are:

- Modifications can be made without changing the wiring and adding devices.
- Minimum occupancy space
- Lower cost of installation
- Economy of maintenance. In addition to increasing the reliability of the system, by eliminating mobile contacts, the same automatons can detect and indicate possible malfunctions.

- Possibility to govern several machines with the same automaton.
- Reduced time to start the process due to reduced wiring time.
- If for some reason the machine is out of service, the automaton is still useful to control another machine or production system.

### *3.1.3.6. PLC on the project*

The PLC used in this project is the DHF41B of SEW Eurodrive. This device will allow the movement of the robot in its different axis. It can control 10 I/O and in this machine it will control the different sensors and motors of the different axis.

This PLC is characterized by a great variety of interfaces and a higher performance level, which allows complex calculations and interpolated movements. It can be used with different fieldbus interfaces which are: Ethernet TCP/IP, PROFIBUS DP-V1, DeviceNet and UDP.

In the device there are five modules: the output interface, which control de outputs of the PLC, the power supply module, which convert AC/DC, and three Moviaxis inverters (one for each axis x, y, z). Each inverter has its own inputs and outputs, some of them with pre-defined function and others user-programmable. For example, the hardware limit switches can be connected to these inputs so the controller will have faster response.

Four cables provide current to all the modules.

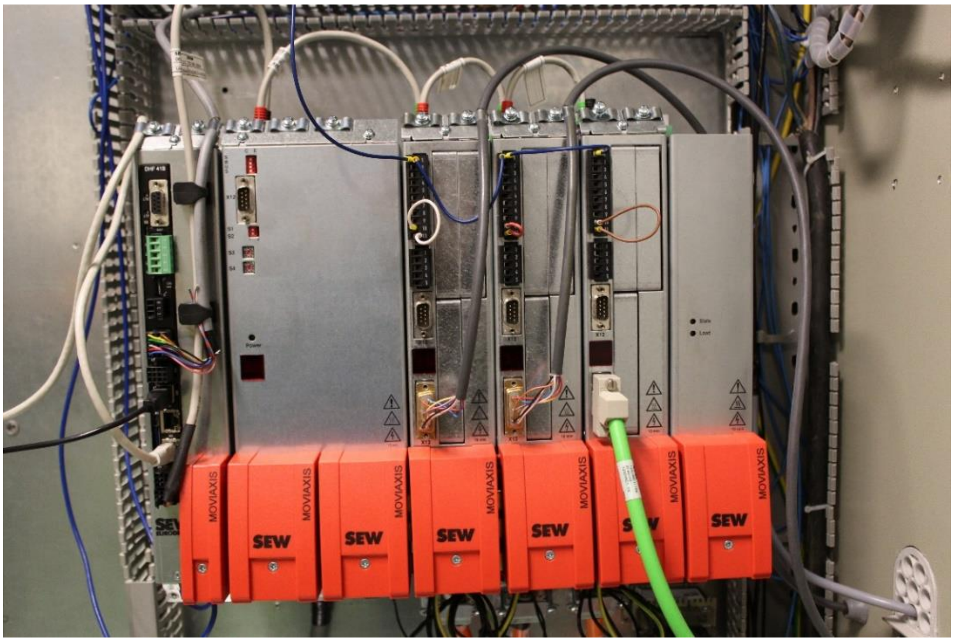


***Image 23.-*** *DHF41B of SEW Eurodrive.*

# 4. Software

The software is the logical equipment of a digital computer, and includes all the legal components necessary to specific tasks to be performed; In contrast to the physical components of the system called hardware that are, those discussed above.

Now, there is the configuration and programming task of the project. Without a good configuration the robot can be dangerous.

## 4.1. Configuration

This section shows the configuration of the data set that determines the value of any of the operating system variables, these options are loaded at the beginning and they couldn't be changed during a process.

During this task, it will be explained how to configure the robot, how to use the manual mode to move the axis of the machine and how to use the program.

### 4.1.1. MoviTools Configuration

As it is said above, the tool used to configure this project is MOVITOOLS MotionStudio of SEW Eurodrive. Following, it will be described, step by step, the actions that must be performed to make a correct configuration of the robot.

1. The first step is to create a *New project* and click on the box ***OK.***


***Image 24.-*** *Create a new project.*

2. Just after that, it's necessary to do a Network Scan to assign a network which will establish a communication channel between the PC and the PLC. In this case we have established the communication through USB.

    - First, activate the online mode. In online work you work directly with the connected units.
    - After that, configure communication connections: is used to set communication interfaces and parameters.
    - And finnaly, start the network scan where you will find the units of the network.

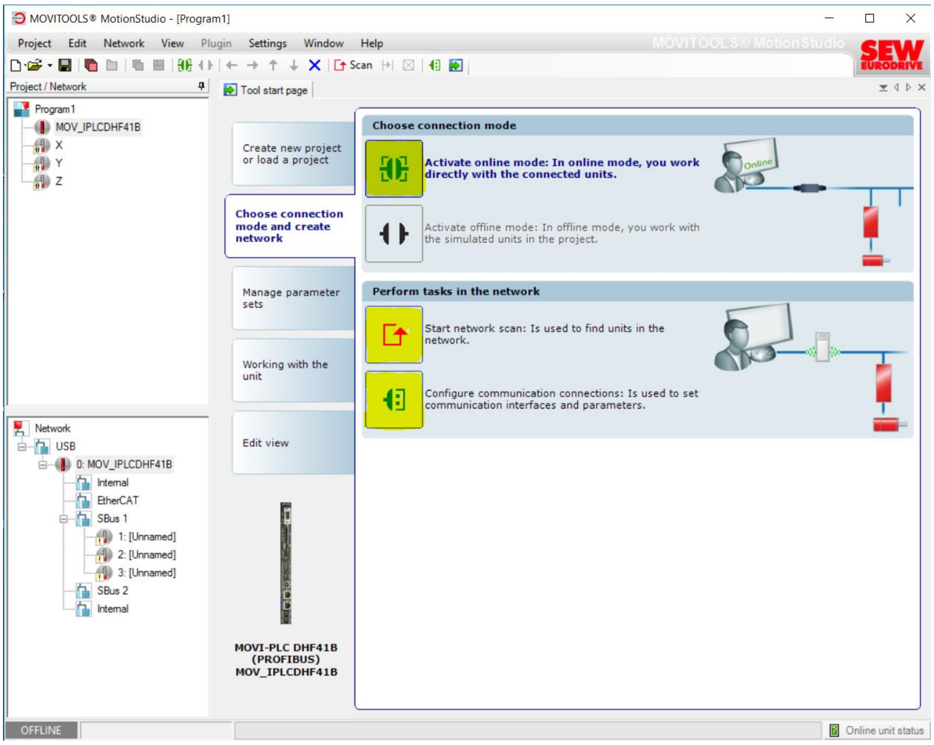



***Image 25.-*** *Switch online mode, Configure communication connections and Network Scan.*

3. Unclick PLC Firmware in *Settings > Options > Other > ...include with MOVI-PLC firmware.*

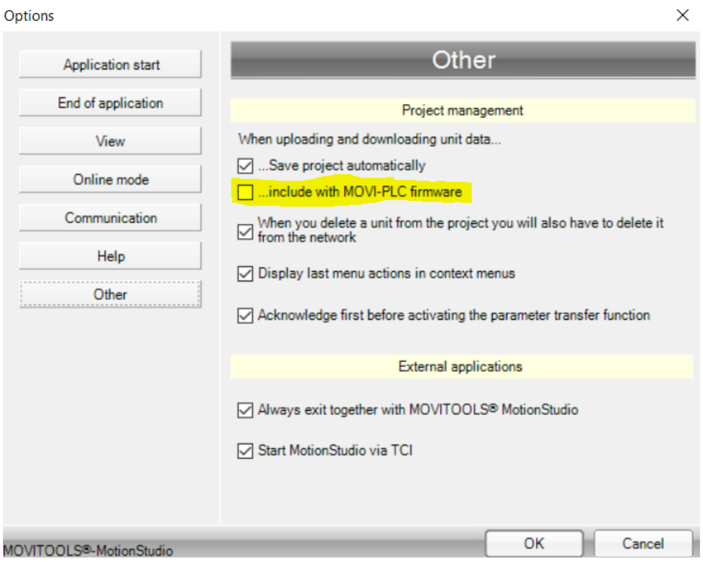



***Image 26.-*** *Unclick PLF Firmware.*

4. Drag all the components to the project window. First with the PLC and then with all the axis individually.

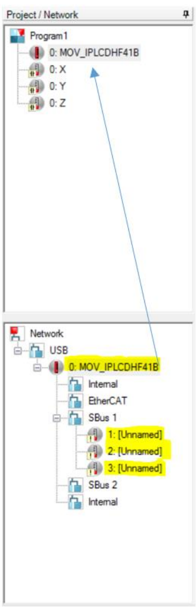



***Image 27.-*** *Drag all the components to the project window.*

5. Make sure to have the PLC updated with the latest version (Version Management).

6. Create a PLC project with the template AxisControl_MultiMotion_Light and import the files TASKCONFIG_HANDLINGKINEMATICS_MM_BUS_LIGHT.EXP and TECHMODULE_HANDLINGKINEMATICS.EXP

    - NOTE: You will find this files in the folder where you create the project *> devices > MOV_IPLCDHF41B > PLCEditor > "file name" > TechModules > HandlingKinematics.*

    After that, compile the project and load it to the MOVIPLC ([Online > login]) and create a boot project.

7. Add SRL file to PLC's SD card with the File System Monitor. The folder to copy is *TechData* and can be found in the folder where the PLC was created.

    - *Devices > MOVI_IPLCDHF41B > PLCEditor > "file name" > TechModules > HandlingKinematics > TechData*

    This folder has to be copied in the memory card.

    - *Devices > MOVI_IPLCDHF41B > ApplicationData > CardImage > User > Default > Data > DHF41B*

## 4.1.2. Axis Startup Configuration

This step configures the right parameters to work properly and be safe. The X and Y axes must be configure the same way while the Z axes is different because it's a cylinder. To configure them you must follow the next steps:

1. Make right click on the axes *Technology editors > DriveStartup for MOVI_PLC / CCU (online)* to configure the different axis and click *Next*. (Do this step with all the axes: X, Y, Z).


***Image 28.-*** *MOVIAXIS Drive startup for MOVI-PLC/CCU.*

2. Press *Delivery State* in the window Device information just the first time that you configure the machine. Once the process is done, it hasn't to be pressed again as it will erase all the configuration done previously. (Do this step with all the axes: X, Y, Z).

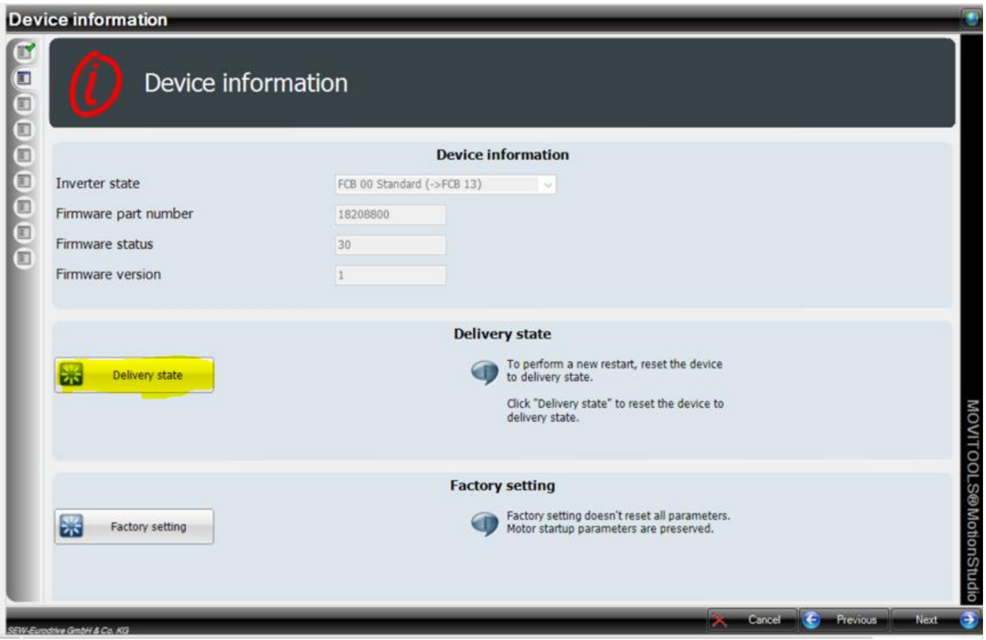



*Image 29.- Device Information.*

3. On the next screen you must select the required startup mode. In that case you must select the *MultiMotion/CCU* option. (Do this step with all the axes: X, Y, Z).

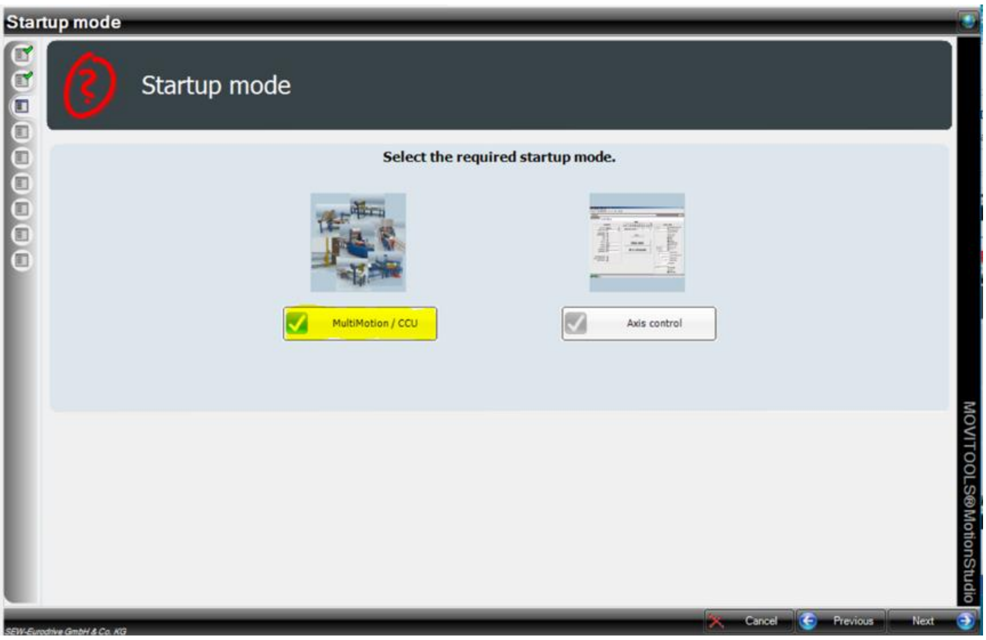



*Image 30.- Startup mode.*

Now it will start the Moviaxis Startup:

1. In the first screen you must select *Parameter set 1.* (Do this step with all the axes: X, Y, Z).

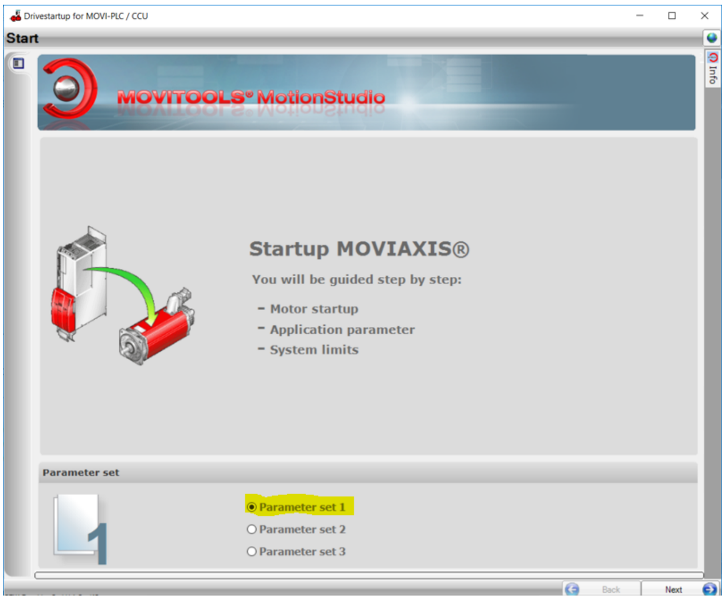



***Image 31.-*** *Startup MOVIAXIS.*

2. The startup mode that has to be chosen for the first time is *Complete startup* where you are going to be able to select the encoder type, chose the motor type and see its information and control the currents limits, moments of inertia, speed and machine. (Do this step with all the axes: X, Y, Z).

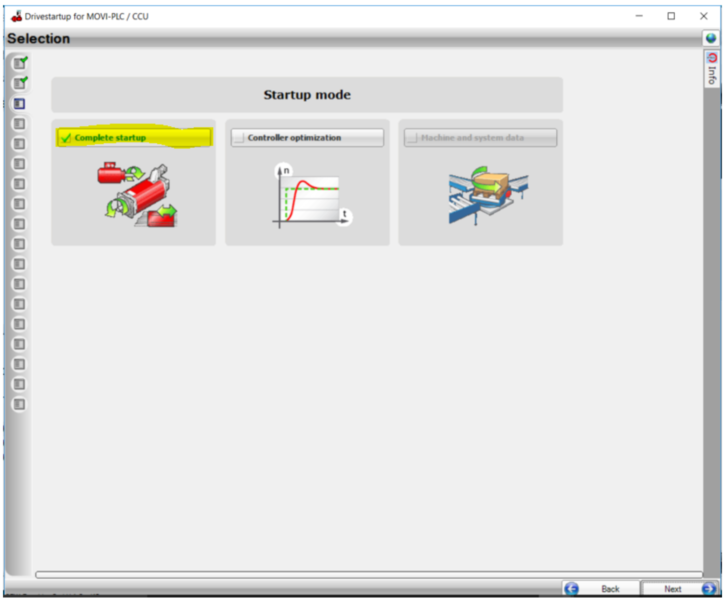



***Image 32.-*** *Startup mode.*

3. Now, it have to be chosen the number of drives. As every inverter is controlling only one motor, the option that have to be selected is *Single-motor operation.* (Do this step with all the axes: X, Y, Z).

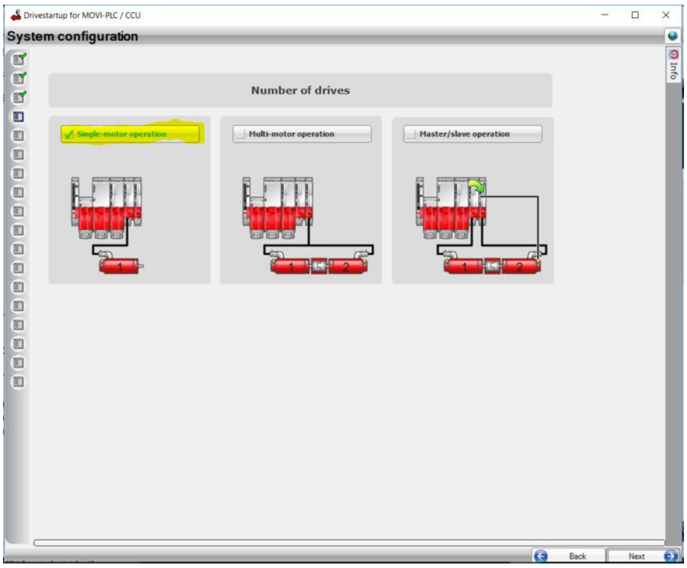



***Image 33.-*** *Number of drives.*

4. The following screen is for configuring the encoder installed in the robot.

- <u>X and Y Axes</u>:

- The relevant encoder grouping that must be select is *Approved encoder.*
- In this machine the encoder installed is a SEW approved encoder with hiperface electrical type, mode: TTK70 manufactured by Sick.

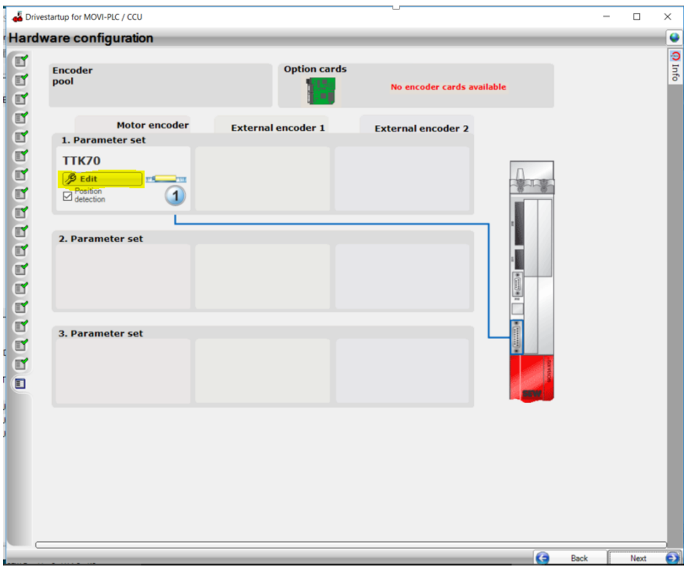



***Image 34.-*** *Hardware configuration.*

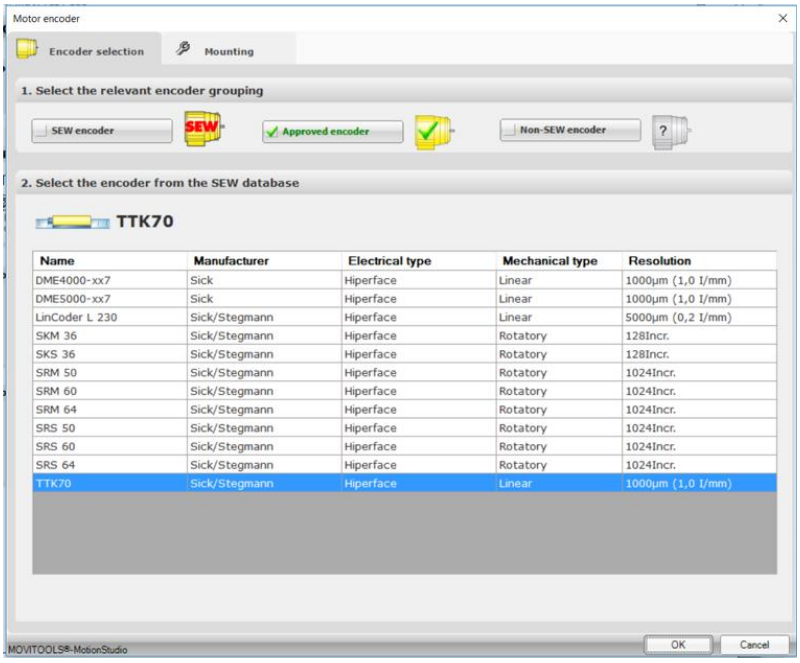



***Image 35.-*** *Encoder Selection.*

- <u>Z Axes</u>:

- You must accept data of the electronic nameplate permanently.
- In this machine the encoder installed is a SEW encoder AK0H.

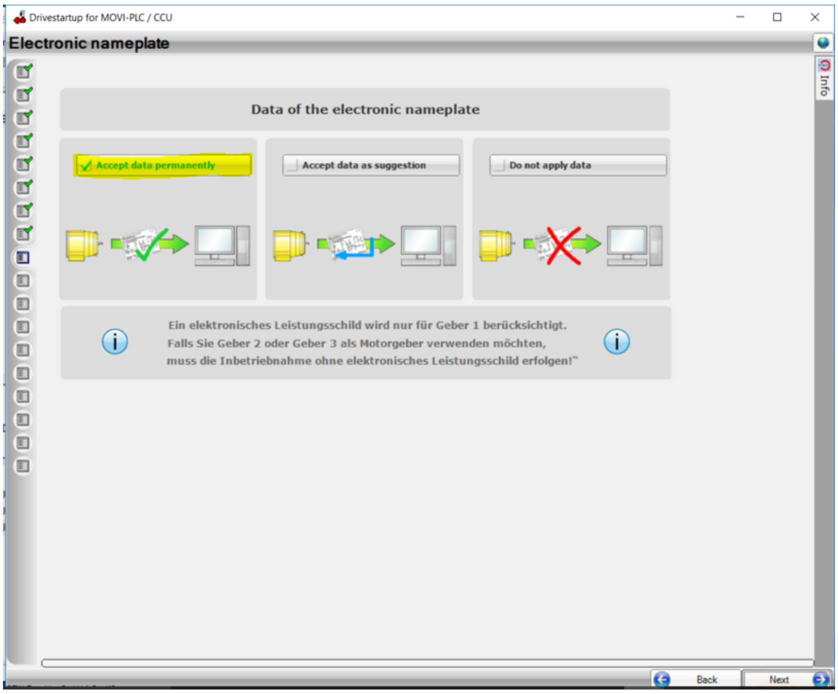



***Image 36.-*** *Electronic nameplate.*

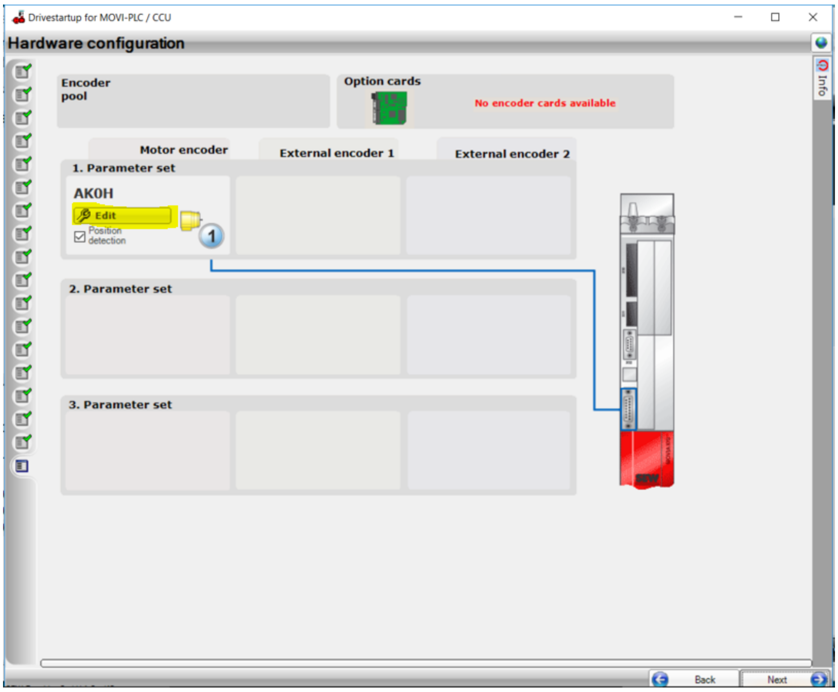



***Image 37.-*** *Hardware configuration*

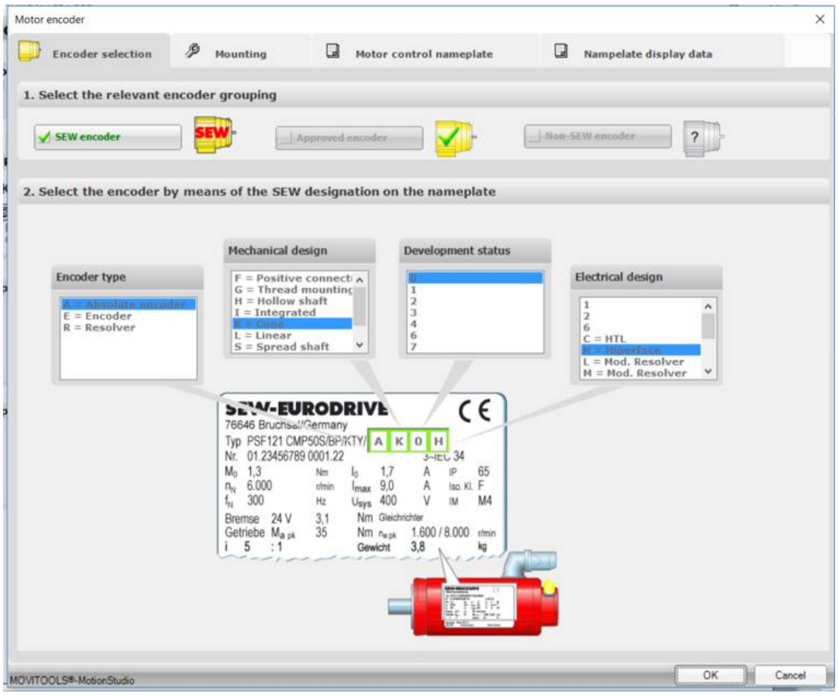



***Image 38.-*** *Encoder selection.*

5. Coming up next, the motor installed must be elected and configured:

   - <u>For X and Y axes</u>: the operating motors are *SL2 linear drives.*
   - <u>For Z axes</u>: CMS electric cylinder.

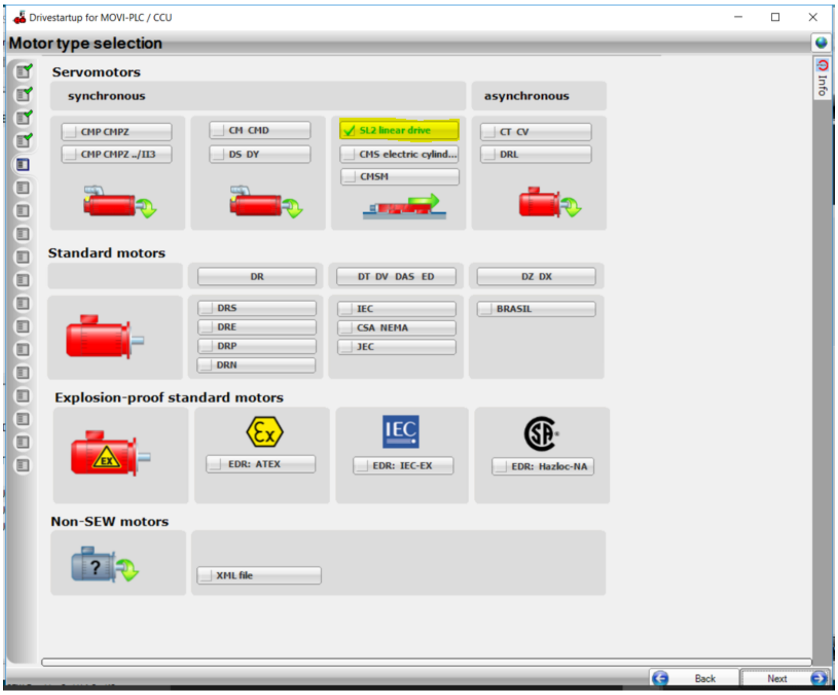



*Image 39.- Motor type selection.*

6. In the upcoming step the model and characteristics of the motor must be selected.

   - <u>For X axes:</u> the model is *SL2-P050VS* with speed of 6000 mm/s; rated main voltage out of the axes = 400 V; Not brake installed; None temperature sensor; Not installed forced cooling fan; and the response to over temperature must be to stop at application limit /waiting.

   - <u>For Y axes:</u> the model is *SL2-P050S* with speed of 6000 mm/s; rated main voltage out of the axes = 400 V; Not brake installed; None temperature sensor; Not installed forced cooling fan; and the response to over temperature must be to stop at application limit /waiting.

   - <u>For Z axes:</u> the model is CM550S with speed of 4500 1/mm; rated line voltage = 400 V; Mounted brake installed; Temperature sensor: KTY; Not installed forced cooling fan; and the response to over temperature must be to stop at application limit /waiting.

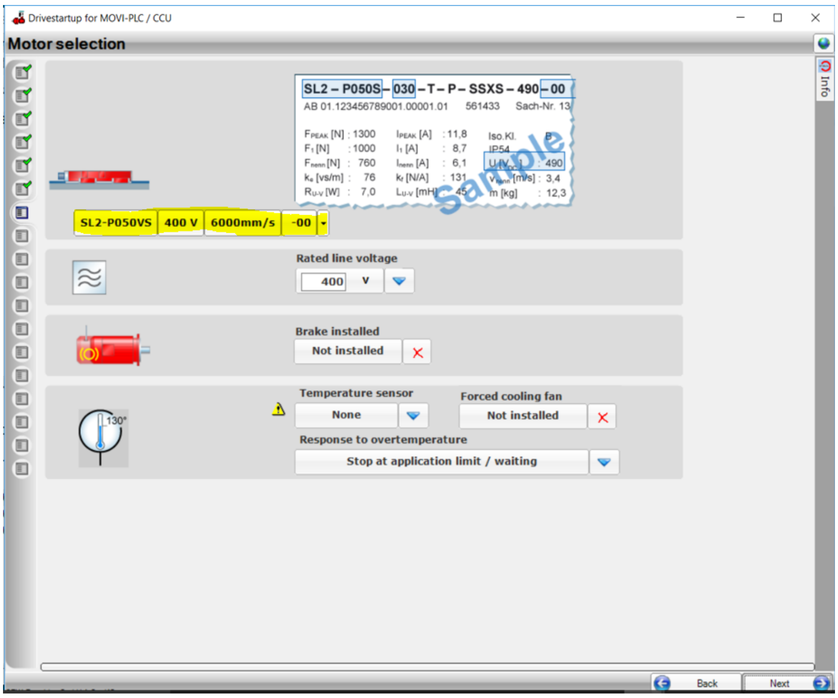



***Image 40.-** Motor selection.*

7. Subsequent, the current limits and delays of SEW suggestion are good enough to accept them. (Do this step with all the axes: X, Y, Z).

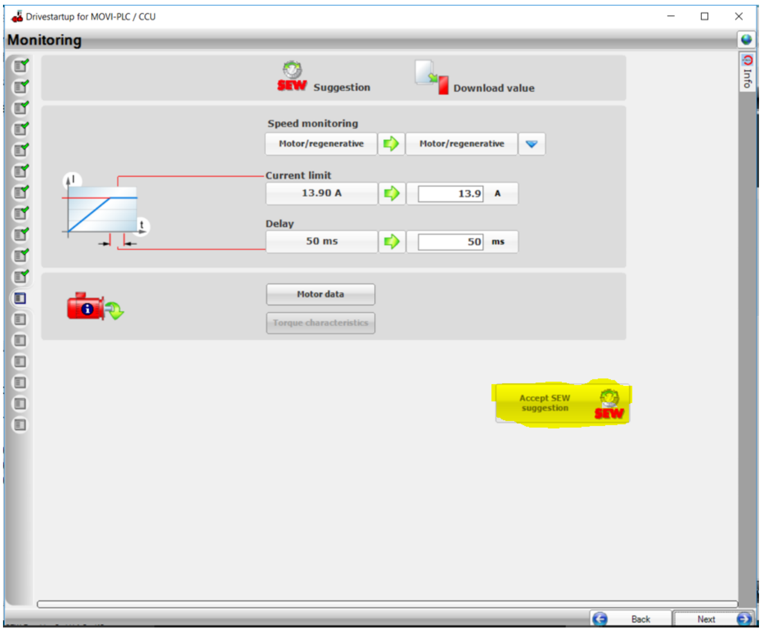



*Image 41.- Monitoring.*

8. The next step is very important, as it has to be configured the load that will support the motors. For the Z axis, there is an approximated load of 30 kg. For the X and Y axis the load is approximately 150 kg. (Do this step with all the axes: X, Y, Z).

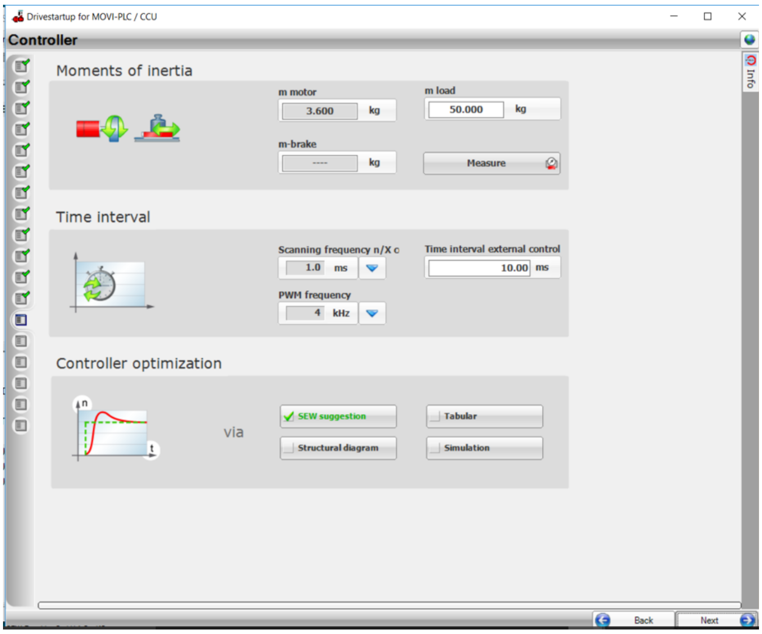



*Image 42.- Controller.*

9. The clearance of load and stiffness depend on the friction that exists in the motor. In this case, as the motors are built with magnets, there is no friction, so these values are optimal. (Do this step with all the axes: X, Y, Z).

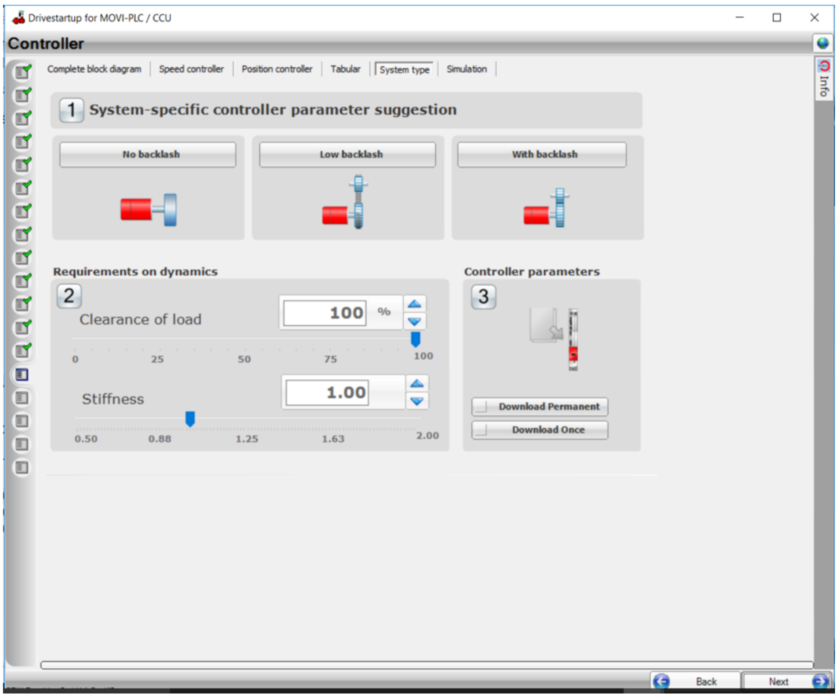



*Image 43.- Controller.*

10. The next step is to load the data into the unit.

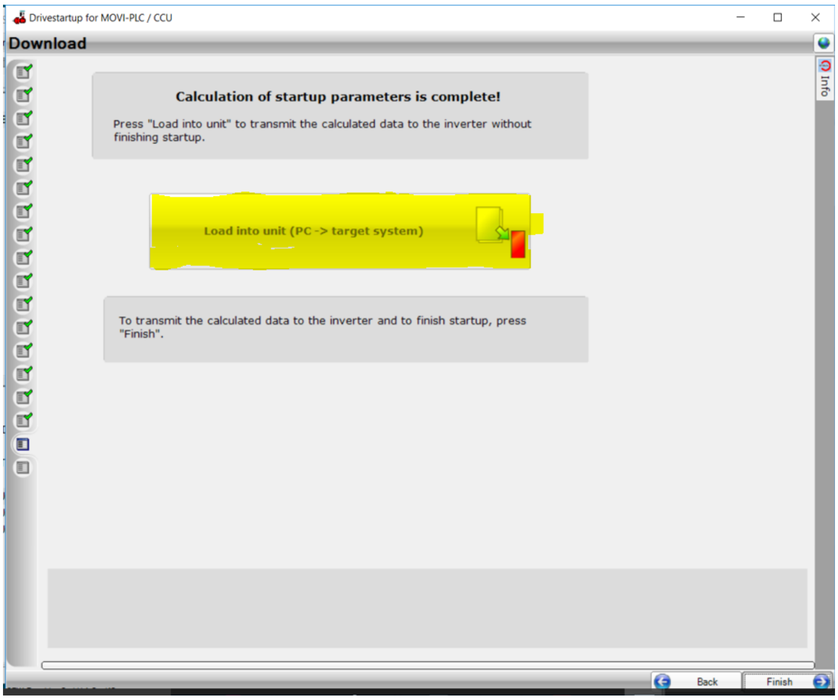



*Image 44.- Load configuration into unit.*

11. When this step is done we have to click finish and then it is time to set the speed and torque limitation values for the motors. It is recommended to set them low and not allow the machine to work at full power to prevent irreversible damage to the structure of the machine.

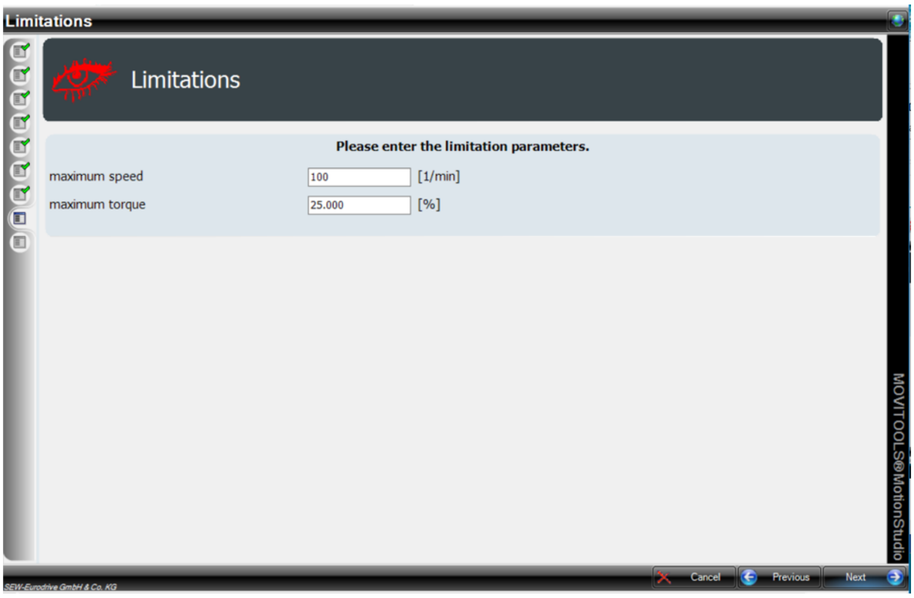



***Image 45.-*** *Limitations.*

12. Once this is done, the final step is to load all the configuration into the device.

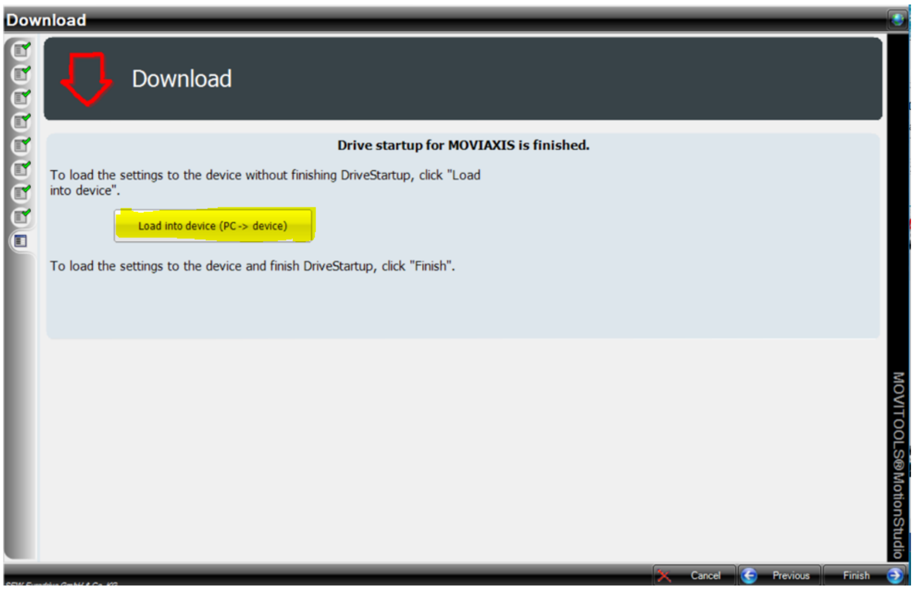



***Image 46.-*** *Load the values configured to the device.*

Now, the motor's sartup is done so the next step is to control them in manual mode. Before that it is necessary to configure the encoder and other parameters so the *Parameter Tree* of each axis must be open:

1. Go to *Parameter Tree > FCB Parameter Settings > FCB 18 encoder adjustment* and set the following encoder parameters:

    - Mode: *Write encoder offset to parameter*.
    - Write control encoder adjustment: *Inactive*.

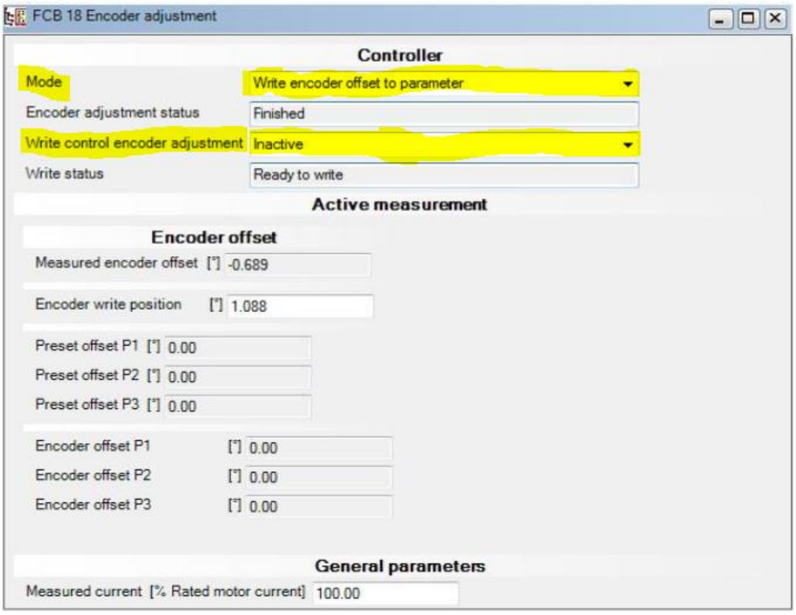



***Image 47.-*** *FCB 18 Encoder adjustment.*

2. Now, a bit of the encoder ust be defined to crry with the encoder adjustment automatically so the moviaxis display will show the number 18.

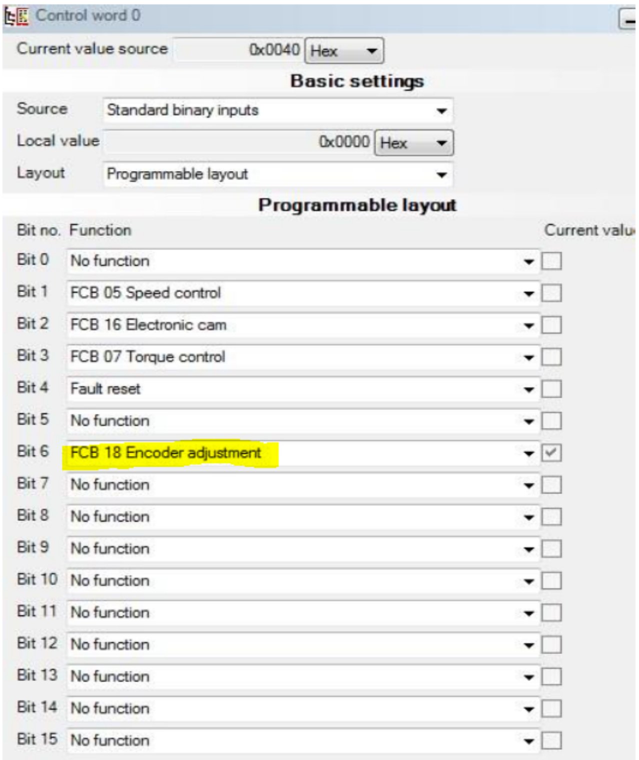



***Image 48.-*** *Control word 0.*

3. The following step, once the calibration has been done in the screen of the *FCB 18 Encoder adjustment* it must appear that:

    - Encoder adjustment statue: *Finished.*

    So the parameters *Write control encoder adjustment* has to be changed to *Write.*

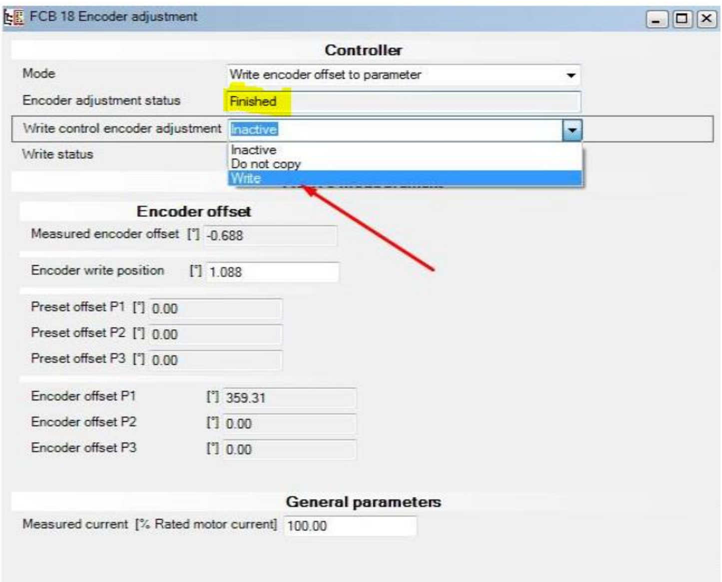



***Image 49.-*** *FCB 18 Encoder adjustment.*

4. The following step is to configure the parmeters related with velocity and acceleration. That has to be made in the *Limit values P1* of the *Parameter Tree.*

    - *System values* have to be about 10% higher than the *Application Limits.*
    - *Emergency stop declaretion* has to be high because then the axes are going to stop really fast in case of an emergency stop.
    - Configure an small torque at the beginning, not allowing the motors to work at full power.

5. Now it is time to set the Moviaxis to mode 09 or positioning, allowing to move the axis to determinate position. That has to be made in the *FCB 09 Positioning 00* of the *Parameter Tree.*

    - *Setpoint local position*: point where the axis will go
    - *Limit values*: maximum speed and acceleration during this movement.

Once all the aformentioned steps are done, we can start the manual mode. To get in the Manual mode window you must press right click on each axis and select *Manual mode (online).*

Click on *Activate manual mode* and it should go on Online mode.

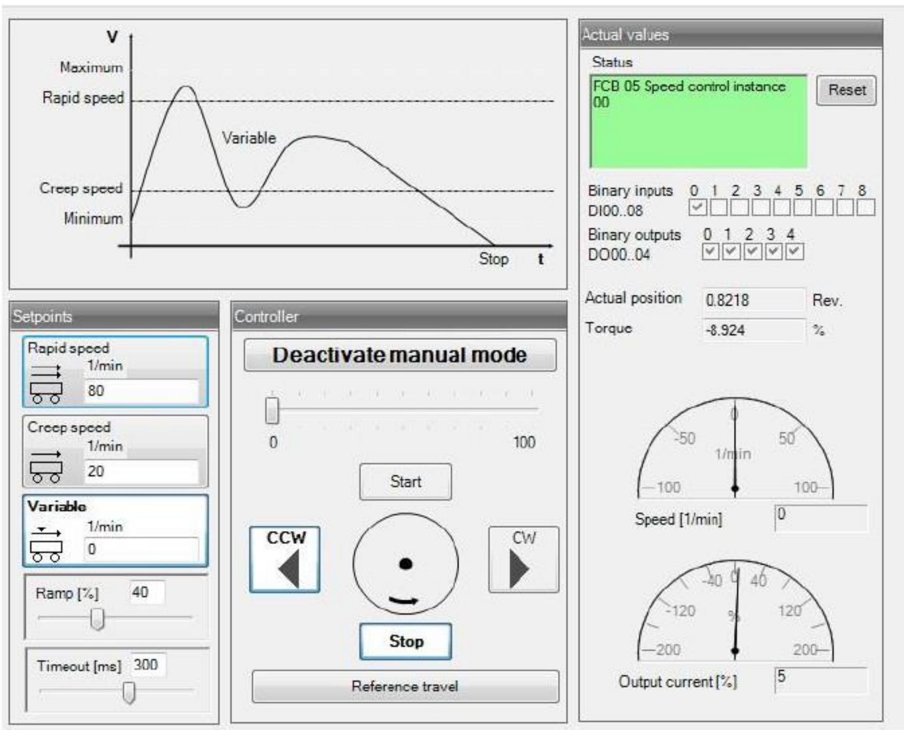



***Image 50.-*** *Manual Mode.*

In my case, the axes X and Y doesn't move in manual mode and some errors apeard. In the following section it will be explained the errors that emerged throughout the project, the solutions tried and the knowledge gained through them.

## 4.2. Programming

Programming is the only part of the robot that is not seen. It is the software that runs the microcontroller, which are the commands that tell the microcontroller what to do.

Now the program structure will be explained:

- The program consist of a Main Task where all the sub-routines are defined. This main program controls all the subprograms and establishes the priority of these.
- Then there are two subroutines:
    - The initialization routine set the initial configuration to be able to run any program. When the routine sets all the parameters activate the variable *Initialization_Done* which will allow the other subroutines to act.
    - The subroutine to move the robot and draw an '8'. This routine will move the axis to different points in a loop thought segments mode.

### 4.2.1. Initialization Routine

This routine basically initialize all the parameters to allow the robot start to an initial parameters values.

When this routine is completed, then the output variable "*Initialization_Done"* is set to '1' and the other programs can read it. This variable is also saved in a intern variable of the function that it is called *"ApplicationReady".*

### 4.2.2. Movement Routine

This routine can be executed once the variable "*Initialization_Done"* it is set to '1'. Then this routine selects the program number and give the position to each segment, which are given in the Main Task program. It also define which will be the last segment setting to '1' the variable "*SegXEnd*" to '1'.

In this case, the program is made to draw a horizontal ‘8’ with the pen of the Z axes. As the robot is able to move two axes at the same time if we move at the same time the axes X and Y to the positions that are set in the program, the robot will draw an ‘8’.

It is needed 13 segments to draw that number.

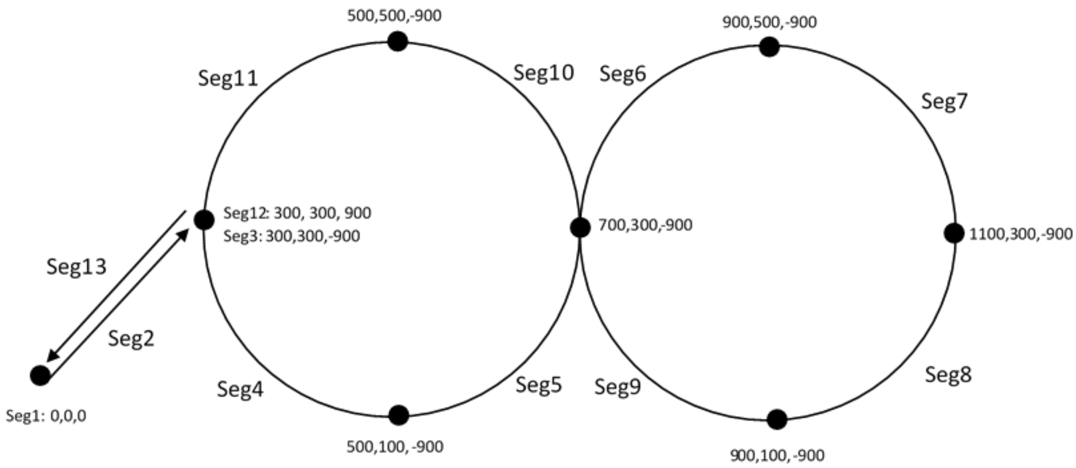



***Image 51.-*** *Segments of the PLC_PRG_Move_To_Segments Routine*

In the Seg12 the axis Z goes up, and in the Seg3 it goes down to start drawing.

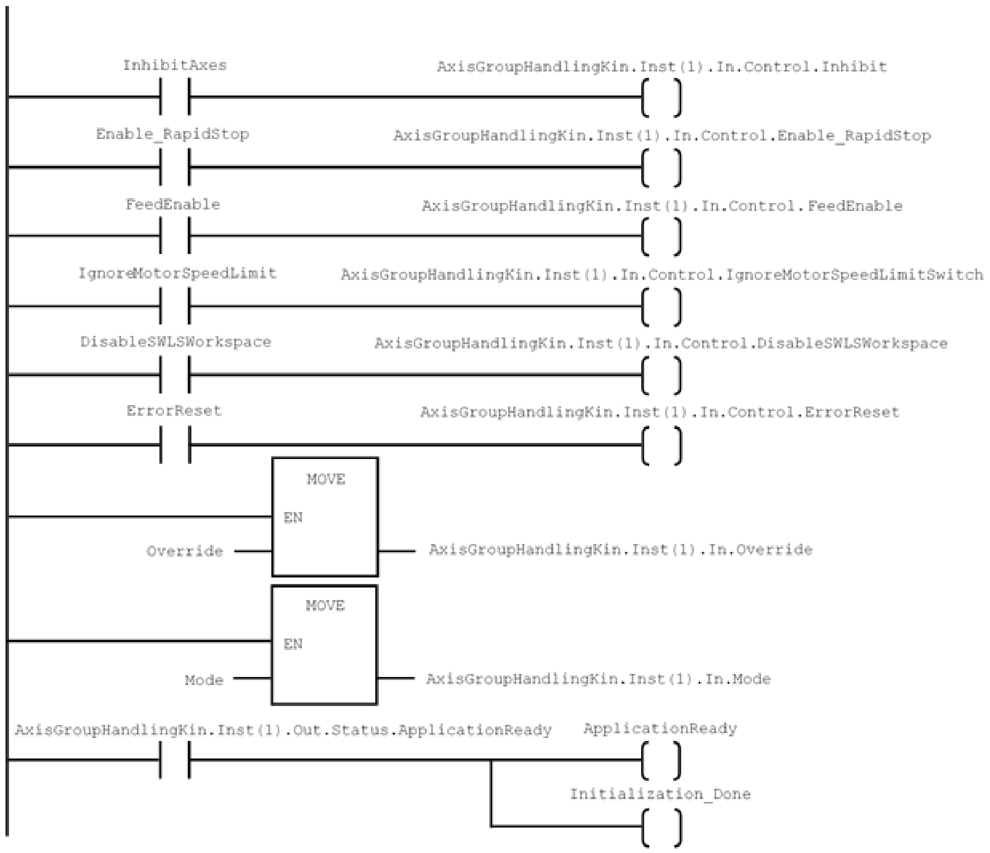
### 4.2.3. Main Program

The Main task call all the other sub routines when it is needs and give them the appropriate inputs.

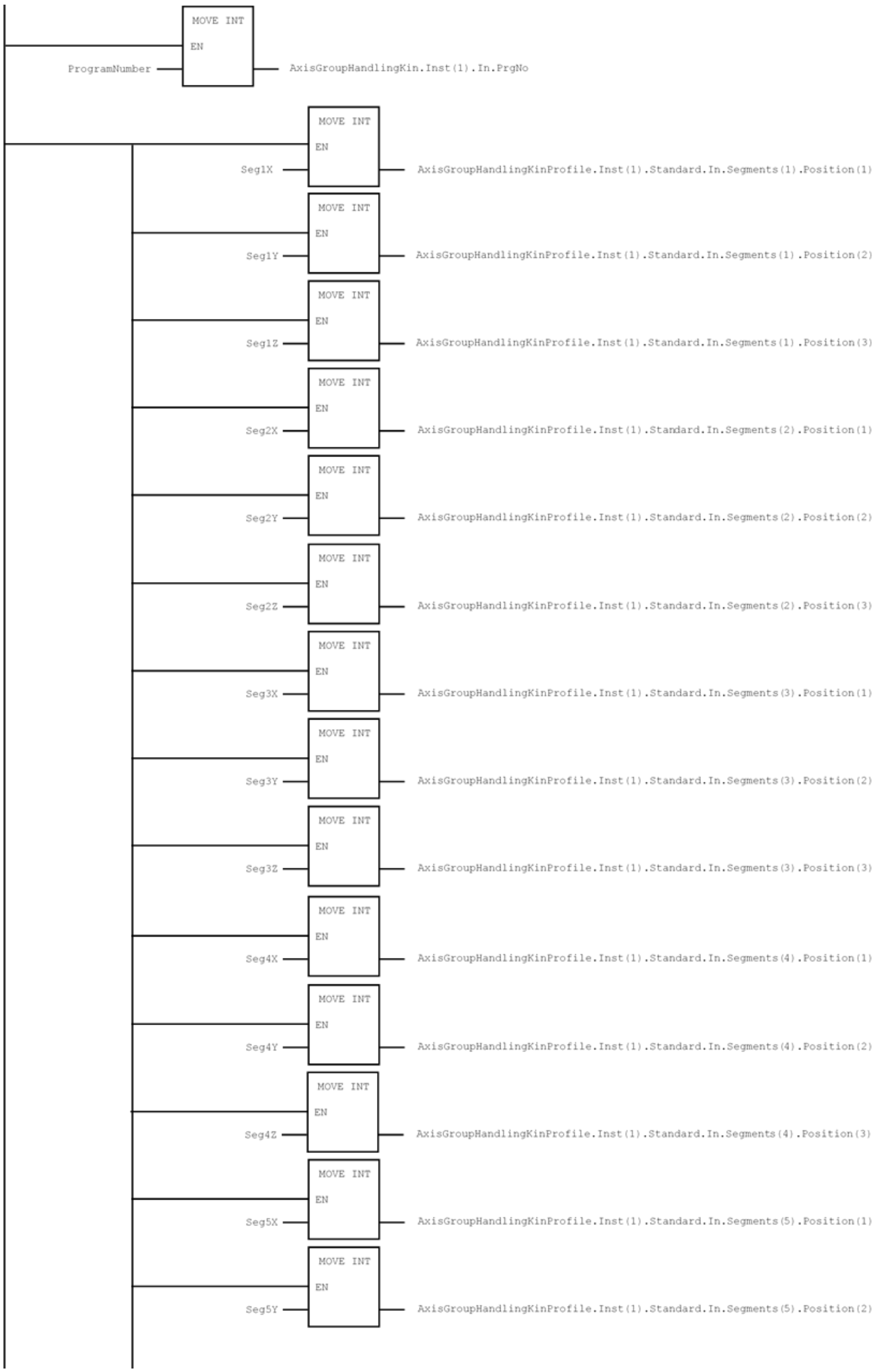
In the following pages there are the program written.

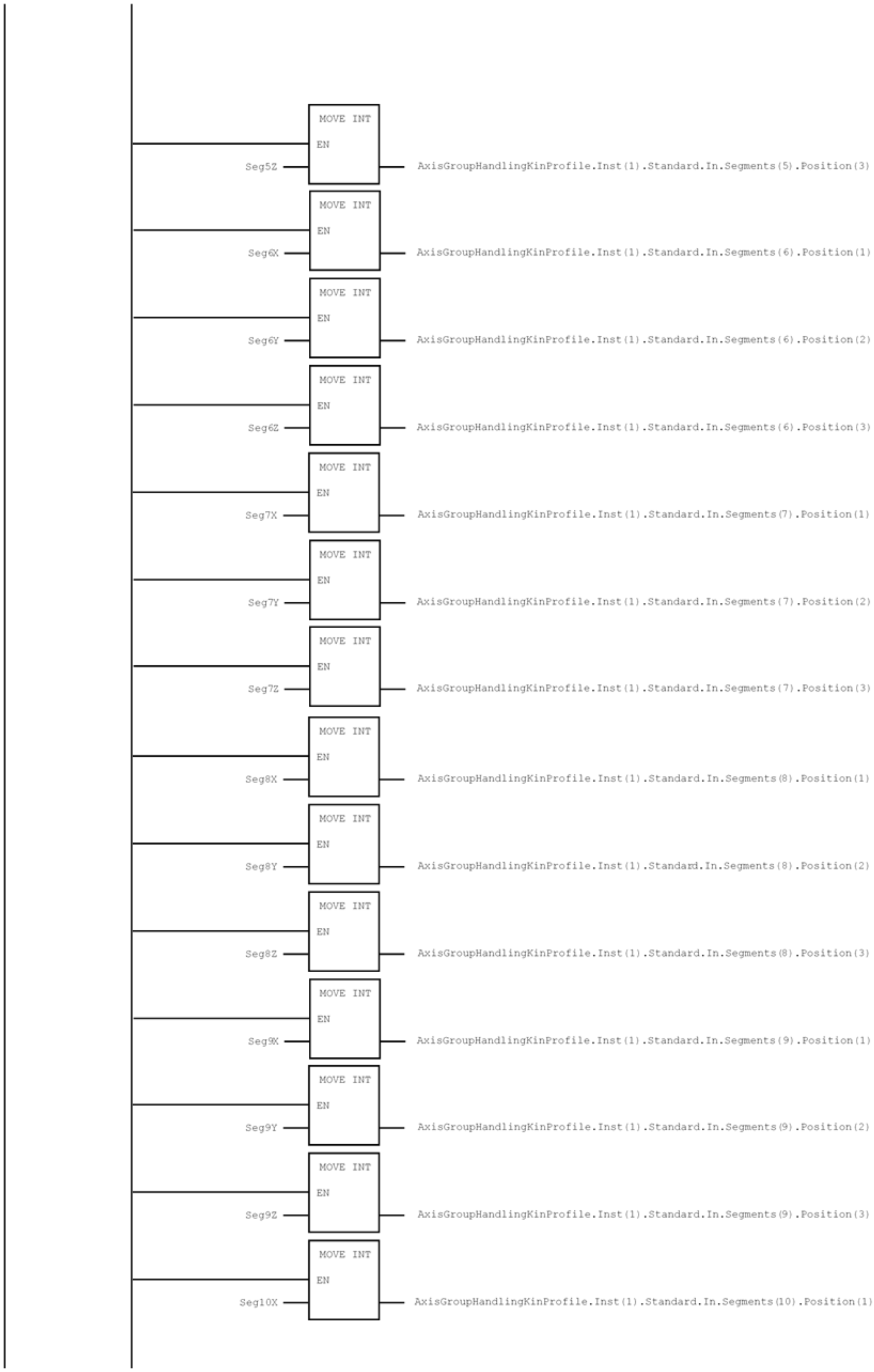
## FUNCTION_BLOCK PLC_PRG_initialize

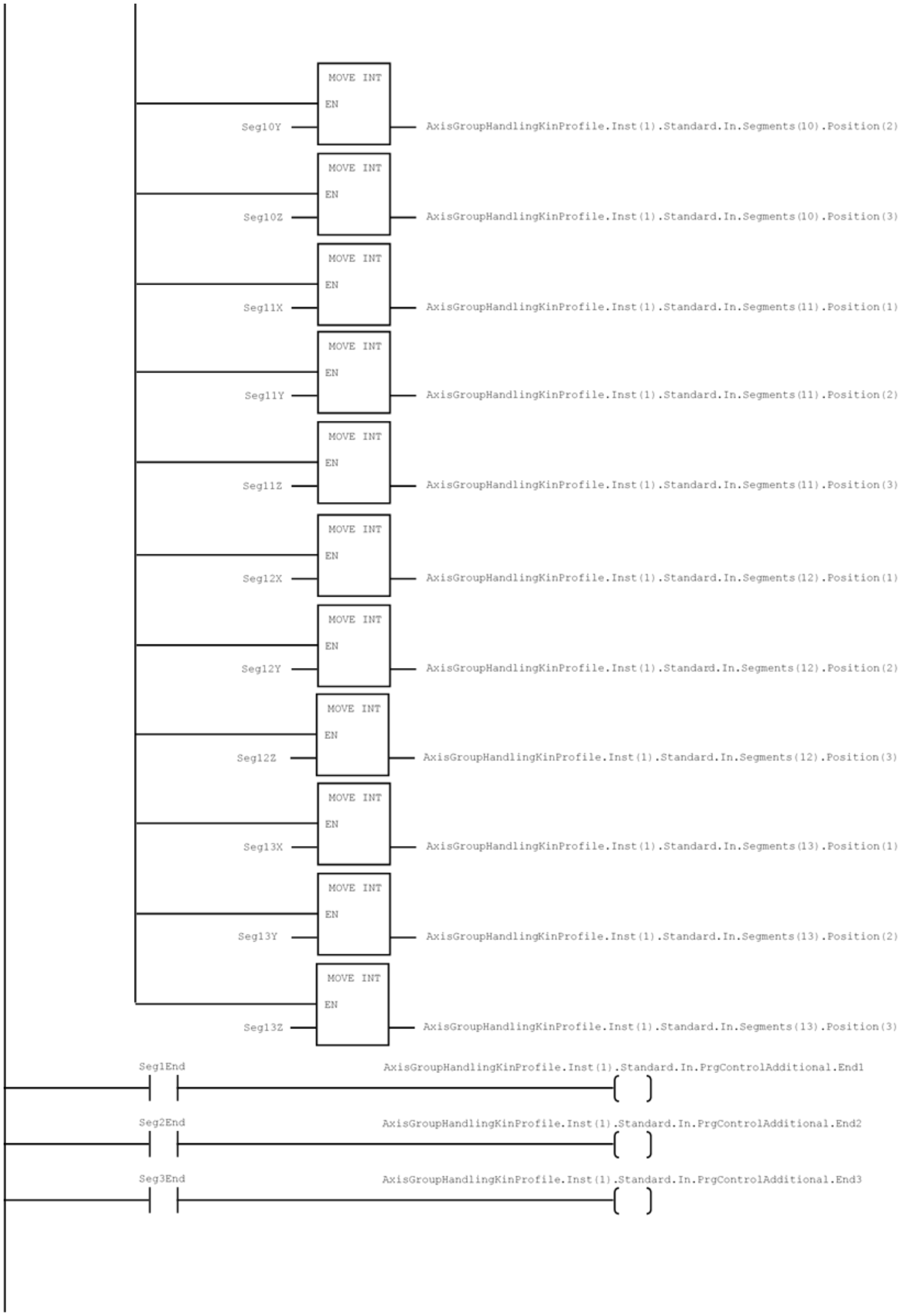
```
VAR_INPUT
    InhibitAxes: BOOL;                  (*has to be '0' to be able to move the axis*)
    Enable_RapidStop: BOOL;             (*has to be '1' to move the axis*)
    FeedEnable: BOOL;                   (*has to be '1' to move the axis*)
    IgnoreMotorSpeedLimit: BOOL;
    DisableSWLSWorkSpace: BOOL;
    ErrorReset: BOOL;
    Override: UINT;                     (*Speed 0-100*)
    Mode: UINT;                         (*Program Mode that will be used for Program Auto
                                        Others Check in Extended Diagnostics for appropriate
                                        number*)
END_VAR

VAR_OUTPUT
    Initialization_Done: BOOL;          (*when Initialization_Done sets to '1' then the
                                        function is completed and the robot is ready to
                                        move*)
END_VAR

VAR
    ApplicationReady: BOOL;             (*Intern variable that save the value of
                                        Initialization_Done variable*)
END_VAR
```

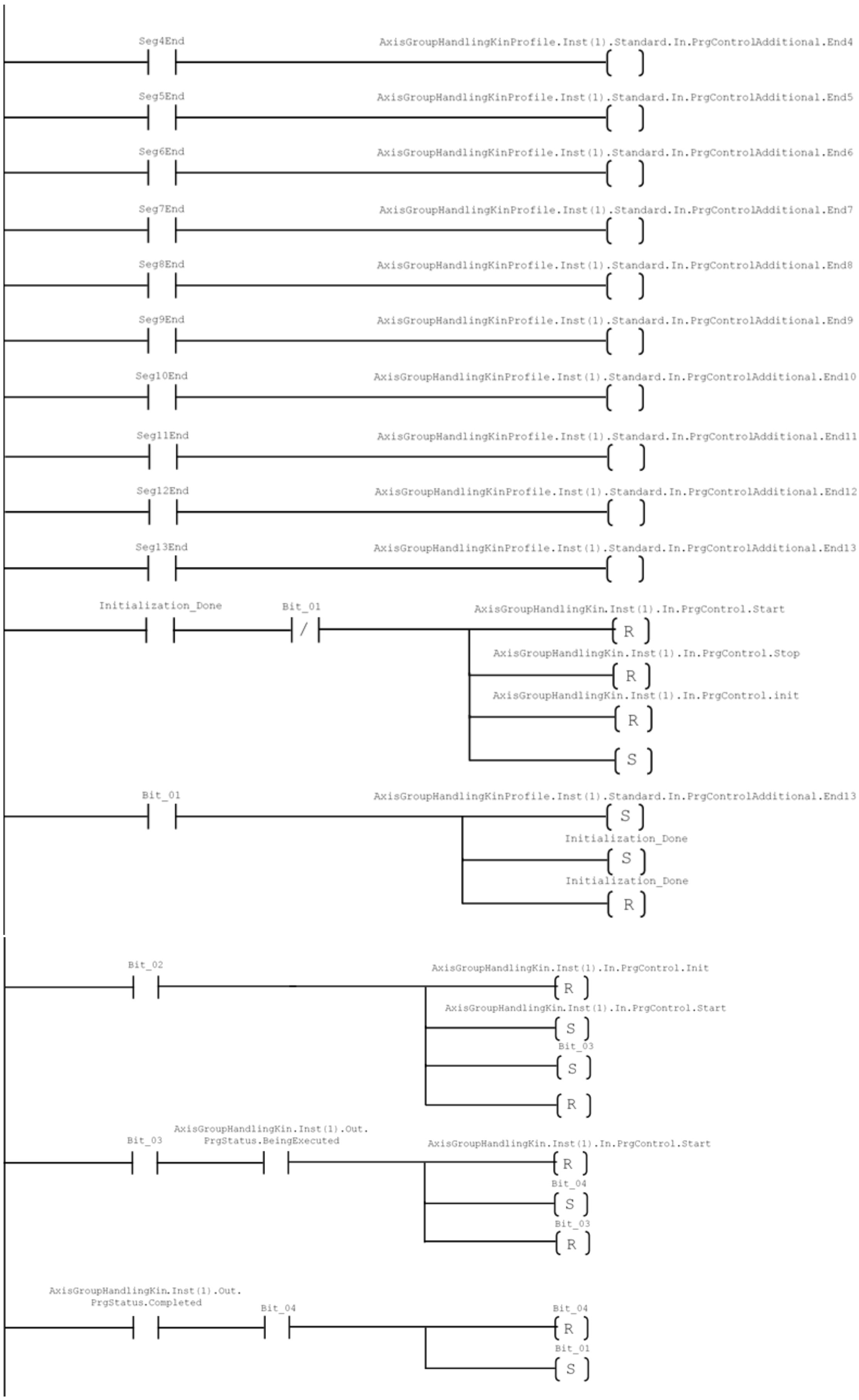


```
FUNCTION_BLOCK PLC_PRG_Move_To_Segments

VAR_INPUT
	Initialization_Done: BOOL;	(*if it's '1' the initialization has been completed
	and the robot is prepared to move*)
	ProgramNumber: BOOL;	(*has to be '1' for Target Axis*)
	Seg1X: INT;	(*position for the segments*)
	Seg1Y: INT;
	Seg1Z: INT;
	Seg1End: BOOL;	(*if it's '1' it ends in this segment*)
	Seg2X: INT;
	Seg2Y: INT;
	Seg2Z: INT;
	Seg2End: BOOL;
	Seg3X: INT;
	Seg3Y: INT;
	Seg3Z: INT;
	Seg3End: BOOL;
	Seg4X: INT;
	Seg4Y: INT;
	Seg4Z: INT;
	Seg4End: BOOL;
	Seg5X: INT;
	Seg5Y: INT;
	Seg5Z: INT;
	Seg5End: BOOL;
	Seg6X: INT;
	Seg6Y: INT;
	Seg6Z: INT;
	Seg6End: BOOL;
	Seg7X: INT;
	Seg7Y: INT;
	Seg7Z: INT;
	Seg7End: BOOL;
	Seg8X: INT;
	Seg8Y: INT;
	Seg8Z: INT;
	Seg8End: BOOL;
	Seg9X: INT;
	Seg9Y: INT;
	Seg9Z: INT;
	Seg9End: BOOL;
	Seg10X: INT;
	Seg10Y: INT;
	Seg10Z: INT;
	Seg10End: BOOL;
	Seg11X: INT;
	Seg11Y: INT;
	Seg11Z: INT;
	Seg11End: BOOL;
	Seg12X: INT;
	Seg12Y: INT;
	Seg12Z: INT;
	Seg12End: BOOL;
	Seg13X: INT;
	Seg13Y: INT;
	Seg13Z: INT;
	Seg13End: BOOL;
END_VAR

VAR_OUTPUT
	Done: BOOL;
END_VAR

VAR
	Bit_01: BOOL;	(*reset everything*)
	Bit_02: BOOL;	(*program initializing*)
	Bit_03: BOOL;	(*program starting*)
	Bit_04: BOOL;	(*program executing*)
END_VAR
```

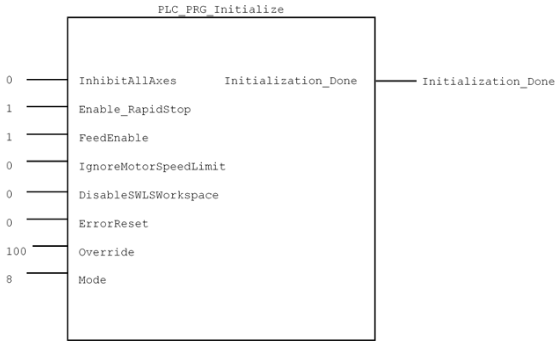

MOVE INT
EN
ProgramNumber
AxisGroupHandlingKin.Inst(1).In.PrgNo
MOVE INT
EN
Seg1X
AxisGroupHandlingKinProfile.Inst(1).Standard.In.Segments(1).Position(1)
MOVE INT
EN
Seg1Y
AxisGroupHandlingKinProfile.Inst(1).Standard.In.Segments(1).Position(2)
MOVE INT
EN
Seg1Z
AxisGroupHandlingKinProfile.Inst(1).Standard.In.Segments(1).Position(3)
MOVE INT
EN
Seg2X
AxisGroupHandlingKinProfile.Inst(1).Standard.In.Segments(2).Position(1)
MOVE INT
EN
Seg2Y
AxisGroupHandlingKinProfile.Inst(1).Standard.In.Segments(2).Position(2)
MOVE INT
EN
Seg2Z
AxisGroupHandlingKinProfile.Inst(1).Standard.In.Segments(2).Position(3)
MOVE INT
EN
Seg3X
AxisGroupHandlingKinProfile.Inst(1).Standard.In.Segments(3).Position(1)
MOVE INT
EN
Seg3Y
AxisGroupHandlingKinProfile.Inst(1).Standard.In.Segments(3).Position(2)
MOVE INT
EN
Seg3Z
AxisGroupHandlingKinProfile.Inst(1).Standard.In.Segments(3).Position(3)
MOVE INT
EN
Seg4X
AxisGroupHandlingKinProfile.Inst(1).Standard.In.Segments(4).Position(1)
MOVE INT
EN
Seg4Y
AxisGroupHandlingKinProfile.Inst(1).Standard.In.Segments(4).Position(2)
MOVE INT
EN
Seg4Z
AxisGroupHandlingKinProfile.Inst(1).Standard.In.Segments(4).Position(3)
MOVE INT
EN
Seg5X
AxisGroupHandlingKinProfile.Inst(1).Standard.In.Segments(5).Position(1)
MOVE INT
EN
Seg5Y
AxisGroupHandlingKinProfile.Inst(1).Standard.In.Segments(5).Position(2)

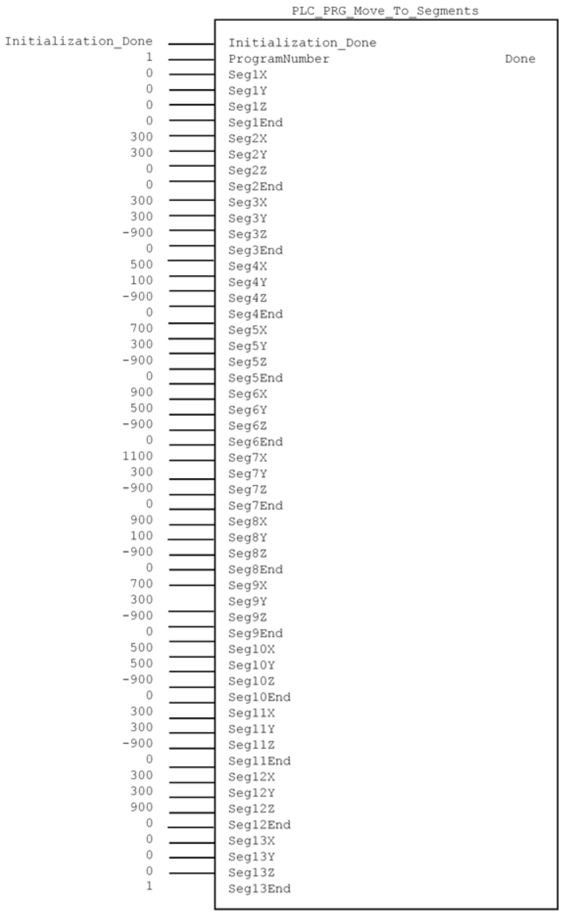

MOVE INT
EN
Seg5Z
AxisGroupHandlingKinProfile.Inst(1).Standard.In.Segments(5).Position(3)
MOVE INT
EN
Seg6X
AxisGroupHandlingKinProfile.Inst(1).Standard.In.Segments(6).Position(1)
MOVE INT
EN
Seg6Y
AxisGroupHandlingKinProfile.Inst(1).Standard.In.Segments(6).Position(2)
MOVE INT
EN
Seg6Z
AxisGroupHandlingKinProfile.Inst(1).Standard.In.Segments(6).Position(3)
MOVE INT
EN
Seg7X
AxisGroupHandlingKinProfile.Inst(1).Standard.In.Segments(7).Position(1)
MOVE INT
EN
Seg7Y
AxisGroupHandlingKinProfile.Inst(1).Standard.In.Segments(7).Position(2)
MOVE INT
EN
Seg7Z
AxisGroupHandlingKinProfile.Inst(1).Standard.In.Segments(7).Position(3)
MOVE INT
EN
Seg8X
AxisGroupHandlingKinProfile.Inst(1).Standard.In.Segments(8).Position(1)
MOVE INT
EN
Seg8Y
AxisGroupHandlingKinProfile.Inst(1).Standard.In.Segments(8).Position(2)
MOVE INT
EN
Seg8Z
AxisGroupHandlingKinProfile.Inst(1).Standard.In.Segments(8).Position(3)
MOVE INT
EN
Seg9X
AxisGroupHandlingKinProfile.Inst(1).Standard.In.Segments(9).Position(1)
MOVE INT
EN
Seg9Y
AxisGroupHandlingKinProfile.Inst(1).Standard.In.Segments(9).Position(2)
MOVE INT
EN
Seg9Z
AxisGroupHandlingKinProfile.Inst(1).Standard.In.Segments(9).Position(3)
MOVE INT
EN
Seg10X
AxisGroupHandlingKinProfile.Inst(1).Standard.In.Segments(10).Position(1)


MOVE INT
EN
Seg10Y
AxisGroupHandlingKinProfile.Inst(1).Standard.In.Segments(10).Position(2)
MOVE INT
EN
Seg10Z
AxisGroupHandlingKinProfile.Inst(1).Standard.In.Segments(10).Position(3)
MOVE INT
EN
Seg11X
AxisGroupHandlingKinProfile.Inst(1).Standard.In.Segments(11).Position(1)
MOVE INT
EN
Seg11Y
AxisGroupHandlingKinProfile.Inst(1).Standard.In.Segments(11).Position(2)
MOVE INT
EN
Seg11Z
AxisGroupHandlingKinProfile.Inst(1).Standard.In.Segments(11).Position(3)
MOVE INT
EN
Seg12X
AxisGroupHandlingKinProfile.Inst(1).Standard.In.Segments(12).Position(1)
MOVE INT
EN
Seg12Y
AxisGroupHandlingKinProfile.Inst(1).Standard.In.Segments(12).Position(2)
MOVE INT
EN
Seg12Z
AxisGroupHandlingKinProfile.Inst(1).Standard.In.Segments(12).Position(3)
MOVE INT
EN
Seg13X
AxisGroupHandlingKinProfile.Inst(1).Standard.In.Segments(13).Position(1)
MOVE INT
EN
Seg13Y
AxisGroupHandlingKinProfile.Inst(1).Standard.In.Segments(13).Position(2)
MOVE INT
EN
Seg13Z
AxisGroupHandlingKinProfile.Inst(1).Standard.In.Segments(13).Position(3)
Seg1End
AxisGroupHandlingKinProfile.Inst(1).Standard.In.PrgControlAdditional.End1
Seg2End
AxisGroupHandlingKinProfile.Inst(1).Standard.In.PrgControlAdditional.End2
Seg3End
AxisGroupHandlingKinProfile.Inst(1).Standard.In.PrgControlAdditional.End3


Seg4End
AxisGroupHandlingKinProfile.Inst(1).Standard.In.PrgControlAdditional.End4
Seg5End
AxisGroupHandlingKinProfile.Inst(1).Standard.In.PrgControlAdditional.End5
Seg6End
AxisGroupHandlingKinProfile.Inst(1).Standard.In.PrgControlAdditional.End6
Seg7End
AxisGroupHandlingKinProfile.Inst(1).Standard.In.PrgControlAdditional.End7
Seg8End
AxisGroupHandlingKinProfile.Inst(1).Standard.In.PrgControlAdditional.End8
Seg9End
AxisGroupHandlingKinProfile.Inst(1).Standard.In.PrgControlAdditional.End9
Seg10End
AxisGroupHandlingKinProfile.Inst(1).Standard.In.PrgControlAdditional.End10
Seg11End
AxisGroupHandlingKinProfile.Inst(1).Standard.In.PrgControlAdditional.End11
Seg12End
AxisGroupHandlingKinProfile.Inst(1).Standard.In.PrgControlAdditional.End12
Seg13End
AxisGroupHandlingKinProfile.Inst(1).Standard.In.PrgControlAdditional.End13
Initialization_Done
Bit_01
AxisGroupHandlingKin.Inst(1).In.PrgControl.Start
R
AxisGroupHandlingKin.Inst(1).In.PrgControl.Stop
R
AxisGroupHandlingKin.Inst(1).In.PrgControl.init
R
S
Bit_01
AxisGroupHandlingKinProfile.Inst(1).Standard.In.PrgControlAdditional.End13
S
Initialization_Done
S
Initialization_Done
R
Bit_02
AxisGroupHandlingKin.Inst(1).In.PrgControl.Init
R
AxisGroupHandlingKin.Inst(1).In.PrgControl.Start
S
Bit_03
S
R
Bit_03
AxisGroupHandlingKin.Inst(1).Out.
PrgStatus.BeingExecuted
AxisGroupHandlingKin.Inst(1).In.PrgControl.Start
R
Bit_04
S
Bit_03
R
AxisGroupHandlingKin.Inst(1).Out.
PrgStatus.Completed
Bit_04
Bit_04
R
Bit_01
S

```
FUNCTION_BLOCK PLC_TaskMain

VAR
        PLC_PRG_initialize: PLC_PRG_initialize;
        PLC_PRG_Move_To_Segments_Loop: PLC_PRG_Move_To_Segments;
        Initialization_Done: BOOL;
END_VAR
```



# 5. Errors during the project

During the project there have been errors that have failed to reach the main objective of the project. Then, in this section, it is explained how the errors appeared along this work have been raised.

When an error appears the first thing that has to be done is the following:

- Understand the problem: First of all it is necessary to understand the problem and the components of the machine that are involved in this.
- Create a plan: Secondly we tried to create a plan to solve this error.
- Carry out the plan: Then we execute the established plan.
- Review and interpret the result: And finally, we draw a conclusion of the result obtained.

## 5.1. Errors

It is a very complex system with an own software of the company that has produced the machinery: SEW Eurodrive.

Mostly all the errors that have been produced during the project have been during the configuration process. Here is when the machine and the computer tried to interact together and in that moment a lot of communication errors raise. Other of the errors have been produced because of a wrong selection of parameters or the missing of some steps during the process.

The machine can't move during the manual move so coming up next, the errors that arose during the project will be cited and we will do a study of these as discussed above.

### 5.1.1. The motor is not enabled

#### *5.1.1.1. Understand the problem*

The fact that the motor isn't enabled provokes that the module doesn't respond to the orders sent from the PC so the robot will never move.

#### *5.1.1.2. Create a plan*

This kind of error can be due to having unconnected the Digital Inputs of the Moviaxis to the 24V power supply so we should check their correct connection.

To solve the problem it is necessary to activate the first digital input which actuates as a safety device.

#### *5.1.1.3. Result and conclusion*

When we activate the first digital input, the controller inhibit signal is active and the motor can be turned on and will be able to be controlled but this isn't the case so we must continue to look for the problem whereby the computer is unable to send instructions to the engines and move them.

### 5.1.2. Speed monitoring error

#### *5.1.2.1. Understand the problem*

This error occurs when the active speed monitoring detect an unacceptable deviation between the set point speed and the actual speed.

#### *5.1.2.2. Create a plan*

It could be an error of the encoder or a problem with the speed limits that's has been configured so the plan is to increase the set point speed limits in the startup and in the parameter tree.

#### *5.1.2.3. Result and conclusion*

The error disappeared for a while but it appear more times so we realized that it's a encoders problem so we must continue to look for the problem whereby the computer is unable to send instructions to the engines and move the axis X and Y.

### 5.1.3. Encoder error

#### *5.1.3.1. Understand the problem*

There is a problem with the information transference between the encoder and the computer, for that reason we try to understand how the encoder works and calculate the position value:

The encoder that is used in the project is a TTK70 of Sick. The TTK70 is a non-contact linear encoder consists of a compact read head and a magnetic tape. The magnetic tape is equipped with a magnetic partition and forms the measurement scale. The partition consists of an incremental and an absolute track (two-track tape). To calculate the absolute position value, the read head detects both the absolute and incremental components. The position value is directly output for further processing.

The TTK70 has an SSI output for absolute positioning and an incremental Sin/Cos output for recording speed in real time.

Different faults have been detected during the commutation search:

- Encoder error (14) when the linear motor is moved manually.
  There are two possible cause that may produce this error: the first one should be that the encoder isn't correctly installed; the second one could be that the encoder isn't correctly connected.

- The linear motor does not start after removal of controller inhibit.
  Motor cable interrupted could be the cause of this problem.

- Encoder error after commutation travel. Compensation movement /1st movement) without noticeable 2nd movement.
  This may occur due to an interrupted motor cable or because an alternating field may be establishing but not a rotating field.

#### 5.1.3.2. *Create a plan*

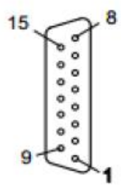
We have been thinking about the errors and their possible causes and the next remedies have been proposed to try to solve them:

- The installation of the encoder will be checked using the encoder manufacturer information:

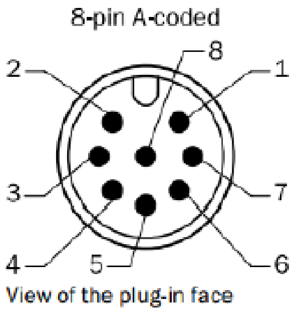
  1. Air gap encoder – measuring tape.
  2. Alignment encoder – measuring tape.
  3. The encoder has to be situated at a maximum of 0,2 mm from the magnetic band, so it is need to make sure that the tolerances are accomplished.
  4. With HIPERFCE encoder: Check mounting direction so that cable output shows in the direction of the smaller absolute values (direction "dot" on the measuring tape if available).

- Check pin assignment, operate HIPERFACE encoder as sin/cos encoder for test purposes:

    - The encoder works with Hiperface connection as following:




| | |
|---|---|
| X13:1 | Signal track A (cos +) |
| X13:2 | Signal track B (sin +) |
| X13:3 | n.c. |
| X13:4 | DATA+ |
| X13:5 | n.c. |
| X13:6 | TF/TH/KTY- |
| X13:7 | n.c. |
| X13:8 | DGND |
| X13:9 | Signal track A_N (cos -) |
| X13:10 | Signal track B_N (sin -) |
| X13:11 | n.c. |
| X13:12 | DATA- |
| X13:13 | n.c. |
| X13:14 | TF/TH/KTY+ |
| X13:15 | $U_S$ |




| Pin | Color of wires | Signal |
|---|---|---|
| 1 | Brown | REFSIN |
| 2 | White | + SIN |
| 3 | Black | REFCOS |
| 4 | Pink | + COS |
| 5 | Gray or yellow | Data + |
| 6 | Green or purple | Data – |
| 7 | Blue | GND |
| 8 | Red | $+U_S$ |
| – | Copper braid | Screen |

***Image 52.-** Encoder connection to Moviaxis Datasheet.*

    - Make sure that the shield of the cables are connected to GND, otherwise the shield would be of no protection at all. The shield has to be in direct contact with the Proactive Earth, so the right way to connect it is just to peel the cable and connect it to the specific places dedicated for that purpose in the control cabinet.

- Check linear motor connection:

  - The linear motors have to be connected to its respective port in Moviaxis:

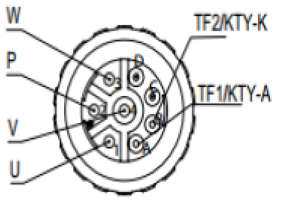
| Plug connector | Pin | Core identification | Assigned |
|---|---|---|---|
| BSTA 078 | 1 | Black with white lettering U, V, W | U |
| | 4 | | V |
|   | 3 | | W |
| | 2 | Green/yellow | PE |
| | A | Black 1 | TF1/KTY-A |
| | B | Black 2 | TF2/KTY-K |
| | C | Black 3 | n.c. |
| View X | D | – | n.c. |

***Image 53.-*** *Linear Motor connection to the Moviaxis Datasheet.*

### *5.1.3.3. Result and conclusion*

- We tested the voltages and currents of the encoder as well as the sensors and everything was correct.
- The connection of linear motor have been checked and it is correctly connected.

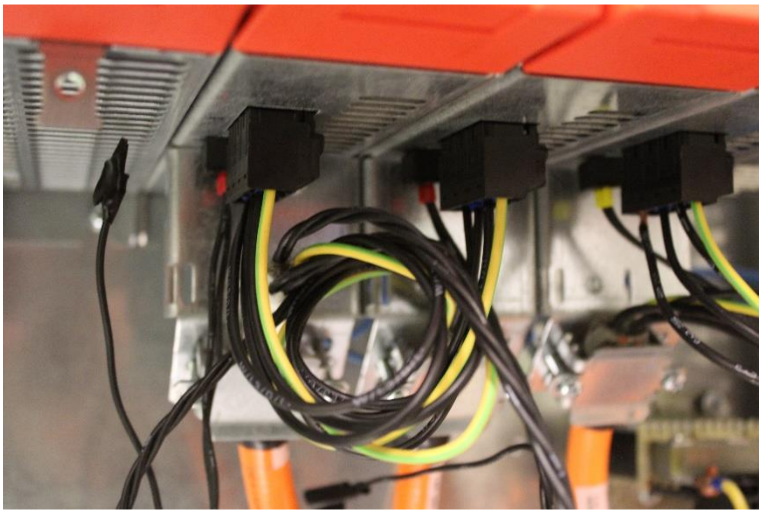


***Image 54.-*** *Connection of linear motor.*

- The connections of the encoder have been checked and it is correctly installed.

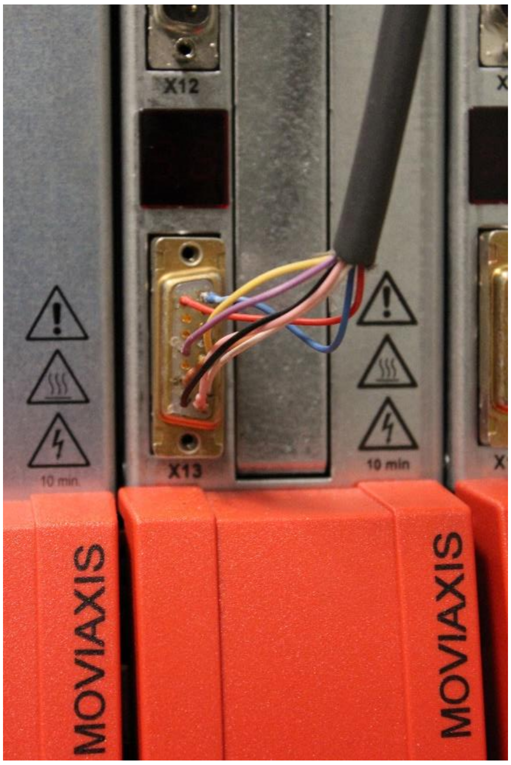



***Image 55.-*** *Connection of the absolut encoder.*

- Encoder configuration have also been checked.
- It was observed how the position increased and decreased when the axis is manually moved and we checked the direction of the encoder.

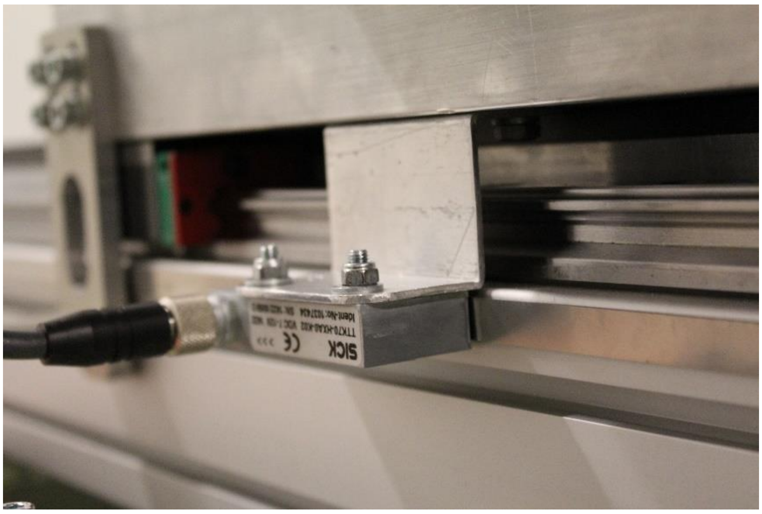



***Image 56.-*** *Absolut encoder.*

Everything seems correct but we still can't perform manual control over the linear motors from the computer so we must continue to look for the error.

As in the time elapsed during the project, it was not possible to find out which is the fault by which we can't control the linear motors of the axes X and Y, here I leave some tips that an expert of SEW-EuroDrive gave to solve the problem but which haven't been put into practice for lack of time.

The tips are the following ones:

- Check the sequence of motor phases.
- Check the measuring tape "arrows" point in the direction of the motor connector.
- Check if the encoder pulses grow when pushing the motor in direction of the connector.
- Scope the encoder pulses along all the measuring tape and check if there are any jumps.

# 6. Conclusion

The main goal of this project hasn't been successfully achieved, which was the realization of a program that allows the robot to draw eights with a linear motor technology and move their axis. The program has been done but it couldn't be simulated or compiled due the axes couldn't be moved.

We all make mistakes, but we do not all face them in the same way. Frustration and fear are the feelings that invaded me when I saw that the things weren't working as expected, but I have learned to use the experience to learn, to get ahead and not to give up. Due to the delivery date, I haven't been able to continue with the project but that doesn't mean that I haven't achieved many other objectives.

Taking part of a real project, made by other team of people, has given me the chance to deal with technical problems that may occur in real-life. A good project gets a lot of time, and usually doesn't work at first.

I learn things of each error I have done, and I learn more of all of them. Now, I perfectly understand how the machinery works. I also learn to use the tool MOVITOOL MotionStudio which I never used before. I have learned to work in a new environment, with new tools, new systems, new people... I have learned to be more versatile.

Working with external professionals from the company SEW-Eurodrive has been a very grateful part of the project where I have learned a lot of thinks about their tools and their machinery.

Concluding, the realization of that project has supposed a challenge due to the large amount of unknown material that is used. Furthermore, it has given me the possibility to put into practice all the skills that I have learned during these 4 years of degree as well as give me a great lesson in the professional field: The only real mistake is the one from which we learn nothing.

# 7. References

[1] *“Sensores inductivos”*. [On-line]. [Last access: 10.03.2017]. Available in: <https://es.slideshare.net/lonely113/sensores-inductivos-8801776>

[2] *“Apunts robotica ari_T1_introducción”*. [On-line]. [Last access: 10.03.2017].

[3] Jordi Alonso Sanchez, Miguel Angel Serra Jaume, ***“Plasma Cutter”.*** Lahti University of Applied Sciences. May 2016.

[4] *“Sensores inductivo”*. [On-line]. [Last access: 10.03.2017]. Available in: <https://es.wikipedia.org/wiki/Sensor_inductivo>

[5] All On Robots. *“Cartesian Robots”*. [On-line]. [Last access: 11.03.2017]. Available in: <http://www.allonrobots.com/cartesian-robots.html>

[6] Universidad Privada Antenor Orrego. *“Scara”*. [On-line]. [Last access: 20.03.2017]. Available in: <https://www.emaze.com/@AWOQIQCT/Robot-Industrial-SCARA>

[7] Luis Suñer. Departamento Electrónica – CIPFP. *“Partes de un robot”*. [On-line]. [Last access: 24.03.2017]. Available in: <http://roboticasunyer.blogspot.fi/2012/12/partes-de-un-robot-elchasis.html>

[8] *“Electrónica”*. [On-line]. [Last access: 28.03.2017]. Available in: <https://es.wikipedia.org/wiki/Electr%C3%B3nica>

[9] Jorge García; Javier Juan. Sfproyectoiprobiotica. *“Robótica”*. [On-line]. [Last access: 28.03.2017]. Available in: <http://sfproyectoiprobotica.blogspot.fi/p/historia.html>

[10] *“Sensores final de carrera”*. [On-line]. [Last access: 28.03.2017]. Available in: <https://es.wikipedia.org/wiki/Sensor_final_de_carrera>

[11] Allen-Bradley. *“Interruptores de final de carrera”*. [On-line]. [Last access: 4.04.2017]. Available in:<http://ab.rockwellautomation.com/es/Sensors-Switches/Limit-Switches>

[12] *“Diferencia entre el PLC y el microprocesador”*. [On-line]. [Last access: 4.04.2017]. Available in:<http://www.ehowenespanol.com/diferencia-plc-microprocesador-hechos_547880/>

[13] Paloma Prieto. 2007. Observatorio Tecnológico Gobierno de España. *“MONOGRAFICO: lenguajes de programación – Principios básicos de PLC”*. [On-line]. [Last access: 5.04.2017]. Available in:

<http://recursostic.educacion.es/observatorio/web/gl/component/content/article/502-monografico-lenguajes-de-programacion?start=2>

[14] *"IIEIA_T1_Introduccion"*. [On-line]. [Last access: 6.04.2017]

[15] *"Programmable logic controller"*. [On-line]. [Last access: 6.04.2017]. Available in: <https://en.wikipedia.org/wiki/Programmable_logic_controller>

[16] IFM. *"IF5539 Datasheet"*. [On-line]. [Last access: 18.04.2017]. Available in: <https://www.ifm.com/de/en/product/IF5539>

[17] EcuRed. 32th may 2017. *"Software"*. [On-line]. [Last access: 19.04.2017]. Available in: <https://www.ecured.cu/Software>

[18] *"Configuración (informática)"*. [On-line]. [Last access: 24.04.2017]. Available in: <https://es.wikipedia.org/wiki/Configuraci%C3%B3n_(inform%C3%A1tica)>

[19] RobotWorx a SCOTT company. *"What are gantry robots?"*. [On-line]. [Last access: 4.05.2017]. Available in: <https://www.robots.com/faq/show/what-are-gantry-robots>

[20] SICK Sensor Intelligence. *"TTK70 Datasheet"*. [On-line]. [Last access: 8.05.2017]. Available in: https://www.sick.com/ag/en/product-portfolio/encoders-and-inclination-sensors/linear-encoders/ttk70/c/g201566

[21] SEW-EURODRIVE. *Product documentation.* [On-line]. [Last access: 10.05.2017]. Available in: <www.seweurodrive.com>